Evolved transistor array robot controllers

Article (Published Version)

Evolved Transistor Array Robot Controllers

Michael Garvie m.garvie@sussex.ac.uk
Department of Informatics, University of Sussex, Brighton, BN1 9QJ, UK

Ittai Flascher ittai@binderr.co
Binderr, Haifa, Israel

Andrew Philippides andrewop@sussex.ac.uk
Department of Informatics, University of Sussex, Brighton, BN1 9QJ, UK

Adrian Thompson adrianth@sussex.ac.uk
Department of Informatics, University of Sussex, Brighton, BN1 9QJ, UK

Phil Husbands philh@sussex.ac.uk
Department of Informatics, University of Sussex, Brighton, BN1 9QJ, UK

https://doi.org/10.1162/evco_a_00272

Abstract

For the first time, a field programmable transistor array (FPTA) was used to evolve robot control circuits directly in analog hardware. Controllers were successfully incrementally evolved for a physical robot engaged in a series of visually guided behaviours, including finding a target in a complex environment where the goal was hidden from most locations. Circuits for recognising spoken commands were also evolved and these were used in conjunction with the controllers to enable voice control of the robot, triggering behavioural switching. Poor quality visual sensors were deliberately used to test the ability of evolved analog circuits to deal with noisy uncertain data in realtime. Visual features were coevolved with the controllers to automatically achieve dimensionality reduction and feature extraction and selection in an integrated way. An efficient new method was developed for simulating the robot in its visual environment. This allowed controllers to be evaluated in a simulation connected to the FPTA. The controllers then transferred seamlessly to the real world. The circuit replication issue was also addressed in experiments where circuits were evolved to be able to function correctly in multiple areas of the FPTA. A methodology was developed to analyse the evolved circuits which provided insights into their operation. Comparative experiments demonstrated the superior evolvability of the transistor array medium.

Keywords

Evolvable hardware, evolutionary robotics, evolved analog circuits, incremental evolution, visually guided behaviour, word recognition, field programmable transistor array, feature selection, evolved visual features, dimensionality reduction.

1 Introduction

Evolvable hardware (EHW), or evolutionary electronics (Thompson, 1998)—the application of evolutionary search algorithms to the design of electronic circuits—developed rapidly during the 1990s into the 2000s, with landmarks including efficient circuit designs evolved in simulation (Koza et al., 1996), the first circuits evolved on real hardware (Thompson et al., 1996), and the first VLSI chip produced exclusively for evolution of electronic circuits at JPL, NASA (Stoica et al., 2000), and continues to flourish today

Manuscript received: 20 June 2018; revised: 20 September 2019 and 13 February 2020; accepted: 21 April 2020.

(Howard et al., 2014; Trefzer and Tyrrell, 2015; Lopez et al., 2014; Ping et al., 2017; Campos et al., 2013). EHW has been applied to a range of practical problems including antenna design (Lohn et al., 2001; Lohn and Hornby, 2006), adaptive pattern recognition (Yasunaga et al., 2000; Cagnoni, 2009), image data compression (Sakanashi et al., 2004), self-regulating and fault tolerant circuits (Lopez et al., 2014; Ping et al., 2017; Campos et al., 2013), and self-checking circuits (Garvie and Thompson, 2003; Garvie, 2005) which outperformed the state of the art in the literature by several orders of magnitude. EHW spawned many highly unconventional circuit designs that operated in very different—often superior—ways to conventional hand designed systems (Thompson et al., 1999; Koza et al., 2003).

However, one EHW research area of considerable early promise that has received less attention in recent years is its application to the real-time processing of sensor data coupled to actuator control in physical systems. Application could include wearable and embedded devices, but traditionally the primary exemplar and testbed for such applications is autonomous robotics. With this article, we hope to reignite interest in that topic by demonstrating for the first time the successful evolution of component level analog electronic circuits for controlling a (physical) robot engaged in visually guided behaviours. By component level we refer to circuits at the level of basic electronic components such as transistors and resistors (in our case primarily transistors).

In fact one of the very earliest works in EHW was the evolution of a hardware control system for a physical robot (Thompson, 1995). A RAM chip-based dynamic state machine (DSM—a kind of generalisation of a finite state machine) was evolved directly in hardware by using a genetic algorithm to configure the RAM contents, with sensors and actuators connected directly to the DSM. Evolved digital circuits for controlling similar, or slightly more complex, robot behaviours soon followed (Keymeulen et al., 1996; Naito et al., 1996; Haddow and Tufte, 2000; Roggen et al., 2003). These later systems were all based around bespoke or proprietary (digital) reconfigurable gate-level circuitry (e.g. FPGAs).

However, an interesting aspect of the original DSM work was the fact that it pointed towards the potential power of evolved *analog* processing in the robotic context. Whether or not the DSM input, state, and output variables were clocked or free-running was genetically determined, as was the rate of the clock for any variables that used it (Thompson, 1995; Thompson et al., 1999). So although the circuit was not exactly analog in the conventional sense, it was not standardly digital either. The potentially rich dynamics of such a system could be exploited to mesh with the dynamics of the robot–environment interactions arising as the robot behaved in the world. Thus tight, efficient sensorimotor loops, running through the environment and the hardware, were evolved, illustrating the power of unconventional electronics unleashed by the EHW approach. This approach chimed with the movements within cognitive science and AI that emphasized dynamical systems understandings of behaviour and behaviour generation in animals and robots (Harvey, 1992b; Beer, 1997; Husbands et al., 1995; van Gelder, 1995).

Thus, when specialized boards that allowed component level evolution of analog circuits started to appear (Layzell, 1999; Zebulum et al., 1998; Thompson et al., 1999; Langeheine et al., 2000, 2002; Stoica et al., 2001), the possible application of such technology to robot control was often discussed (Sekanina and Zebulum, 2005). The potential dynamics of unconventional evolved analog circuits is particularly rich and evolved systems have the potential for online adaptation allowing recovery from faults (Thompson et al., 1999; Garvie and Thompson, 2003). This, combined with insights from

evolutionary robotics (Nolfi et al., 2016; Vargas et al., 2014) where it was discovered that dynamically complex neural networks are highly evolvable (Husbands et al., 2010; Beer and Williams, 2015), suggested that analog EHW might be very well suited to evolving compact controllers operating with small numbers of components, even for visually guided behaviours which traditionally employed high levels of processing (Bekey, 2005). An interesting property of the dynamically complex evolved networks mentioned above is their ability to cope with noisy, poor quality sensory data, even when the networks have very few nodes (Husbands et al., 2010). This suggests that analog EHW might also be a useful approach for low cost real-time hardware applications requiring cheap sensors and simple circuits. Such areas include some applications of active wearable sensors (Mayol et al., 2002; Yang et al., 2017), embedded visual sensors (Bo et al., 2014), low cost consumer robotics and autonomous toys. In this article, we explore an autonomous robotics application, but in such a way (mainly through the deliberate use of low-grade sensors) that the study has some relevance to these other potential applications.

There were considerable technical challenges in applying these early analog EHW boards to robotic control, so no such work was attempted. More recently, Berenson et al. (2005) used field programmable analog arrays (FPAAs) as a substrate for the evolution, in hardware, of artificial neural networks (ANNs) for robot control. They were able to successfully evolve ANN controllers for legged locomotion in bipedal and quadrupedal robots, with evaluations occurring directly on the robots. The authors of that work reasoned that evolving at the lower, component level of analog circuits, rather than with the analog implementation of a neural network, would probably be too difficult within their methodology. The assumption was that the considerably expanded search space would likely mean that the time needed for on-robot evaluations before finding a good solution would become infeasible. Technical constraints on the commercial FPAAs used would have also created challenges for circuit-level evolution. Hence, to date there has been no investigation of component-level analog EHW applied to robot control, or any other application involving real-time coordination of sensory and motor processing in a physical system acting in the world.

Here we fill that gap by presenting the results of the first such investigation. We used a field programmable transistor array (FPTA) to evolve transistor level analog circuits to control a physical mobile robot engaged in various autonomous visually guided behaviours. We deliberately use a low-grade camera to explore the ability of analog EHW to deal with noisy, unreliable visual data. By coevolving visual features along with the analog circuits, feature extraction and selection and dimensionality reduction is automatically integrated into our approach. An incremental approach was used whereby the robot tasks become increasingly more challenging.

Recently there has been renewed interest in analog EHW in other application areas (Trefzer and Tyrrell, 2017; Howard et al., 2014) and of the use of other evolvable (continuous) physical substrates (including in simulated robot control) (Mohid et al., 2016; Miller et al., 2014; Adamatzky, 2013), which also suggests that the time is ripe to re-examine analog EHW in autonomous robotics.

A number of challenges had to be overcome that had perhaps been holding back research in analog EHW approaches to evolutionary robotics. Commercially available FPAAs tend to have relatively small numbers of reconfigurable cells and inputs and outputs, and strong constraints on access to low-level configuration. This renders them of limited use in EHW research, especially in robotic application where more than a handful of sensor inputs may be required (using vision opens up a much more useful and

interesting space of behaviours but requires higher numbers of inputs). A small number of research FPTAs were built (Langeheine et al., 2002; Stoica et al., 2001) but most of these are more than a decade old and rely on outdated software that is not easily compatible with most current computer systems. Somehow such hardware must be made to talk to a mobile robot and accept enough inputs to allow more than the most minimal of visual sensing. In addition, there is the challenge of either evaluating behaviours on the robot within a reasonable timeframe, or developing sophisticated enough simulations to allow general visually guided behaviours to evolve that transfer into the real world without loss in performance. While a number of approaches to this latter problem have been developed over the years (Jakobi, 1998; Nolfi et al., 2016), general simulation techniques that cross the reality gap (Jakobi et al., 1995) for any but the most low resolution cameras have proven elusive (Nolfi et al., 2016; Vargas et al., 2014).

In this article, we describe how these issues were addressed, enabling us to use an FPTA to evolve unconventional transistor circuits that successfully controlled a mobile robot engaged in various visually guided behaviours, including one involving responding to voice commands as well as camera and other sensor inputs. In order to achieve this we developed a new method for simulating the robot in its environment. This allowed us to evolve controllers offline by evaluating the robot in a simulation connected to the FPTA. The controllers then transferred seamlessly to the real world. The circuit replication issue was also addressed in experiments where control circuits were evolved to be able to function correctly in multiple areas of the FPTA rather than just a single area. In order to investigate a wider set of sensory modalities, along with the integration of multiple evolved circuits, FPTA circuits capable of voice recognition were also evolved. These were used in conjunction with the controllers to enable voice control of the robot, triggering behavioural switching. An analysis technique was developed to gain insight into the operation of the evolved circuits.

The central question being explored in this research can be simply stated. Can transistor level analog circuits be evolved to act as controllers for physical robots engaged in nontrivial visually guided behaviours? In this article, we show that the answer is yes.

The secondary questions we were interested in relate to whether the evolutionary process could exploit the dynamics of the analog EHW chip, in conjunction with the robot–environment dynamics, to produce robust behaviour even with noisy, uncertain low-grade visual sensors. We show that it can, making use of an integrated evolutionary visual feature extraction and selection method.

After describing our methodology and experimental setup, we present the results of a series of experiments where progressively more complex behaviours are evolved in an incremental way. Some of the evolved systems are analysed in detail. We also present a set of comparative experiments that dig deeper into which properties of the FPTA make it a reliable evolvable hardware medium. The article finishes with a discussion of our findings and reflections on future directions.

2 Methods

In this section, we describe the core methods and experimental setup used throughout this research. We tackled the problem of evaluating very large numbers of robot controller solutions in the real world by developing new methods for accurately simulating the robot acting in its environment. This allowed us to evolve behaviours using a robot simulation which successfully transferred to the real robot.

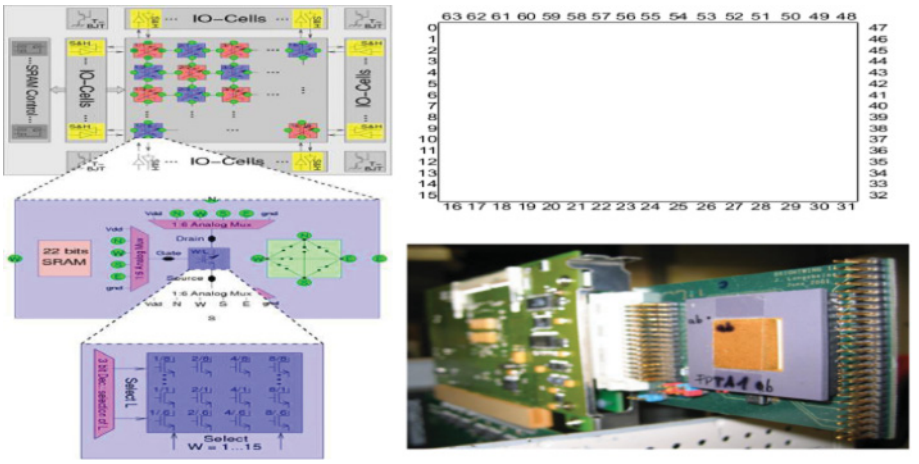

Figure 1: Clockwise from left: Schematic of FPTA at chip, cell, and virtual transistor levels; I/O block numbering; FPTA and host FPGA board. Schematic and photo used with permission by the Electronic Vision(s) Group, Kirchhoff Institute for Physics, Heidelberg University.

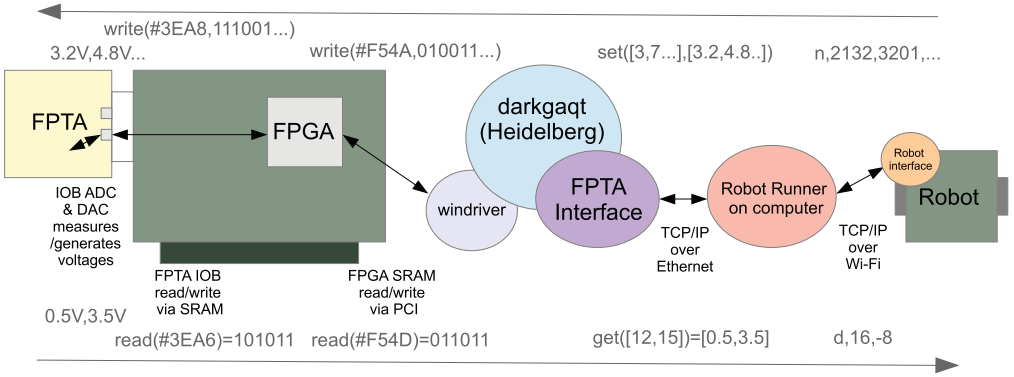

Figure 2: FPTA to robot communication with example IR signals travelling from robot to FPTA (top) and motor actuation signals from chip to robot (bottom). During evolution the GA and simulator replace the Robot Runner and Robot on the right.

2.1 The FPTA

All experiments used the Heidelberg FPTA (Langeheine, 2005) as the underlying evolvable system. This FPTA is a 16 × 16 array of virtual transistors with configurable local routing (see Figure 1, left). Each virtual transistor has a configurable channel width and length. Routing is such that cells can be bypassed and it is impossible to route ground to Vdd (5 V) without going through a transistor. All 64 edges (Figure 1, top right) of the FPTA contain I/O blocks (IOBs) which can be configured for buffered/direct input/output. In buffered I/O mode, DACs and ADCs are used to generate and read voltages from the chip and transfer them back to software (see Figure 2) through a host FPGA board. The FPTA (manufactured using a 0.6 μm triple metal double poly CMOS

process) and the host board rear can be seen in Figure 1 (lower right). The aim was to evolve transistor circuits on the chip to control a mobile robot by feeding robot sensor readings into the chip as inputs and using chip outputs as robot actuator signals. The configurable chip allows us to evolve the individual transistor properties (width and length, which determine the current–voltage characteristics), and the way in which the transistors are connected together on the array. Sensor inputs and motor outputs were mapped to/from the native chip IO range [0,5 V]. The full range of any given sensor was linearly mapped onto the range [0,5] and the chip motor outputs were linearly mapped from [0,5] to the robot's motor range [−20,20] such that 0 mapped to full speed forward (20) and 5 to full speed backward.

2.2 The FPTA Interface

The FPTA is hosted on a PCI FPGA board and communicates via a third party library called Windriver to the Heidelberg "darkgaqt" C++ software. A socket server extension was added to this software providing a JSON-RPC API into the chip allowing FPTA access from any programming language and host through a simple API (*program* chip, *get* and *set* pin voltages). During evolution the interface is accessed by our Java-based custom robot simulator and evolutionary search software. At other times it is accessed by tools to visualise individual solutions or by the Robot Runner which facilitates live robot-FPTA Wi-Fi communication allowing transfer to and fro of sensory inputs and motor outputs (Figure 2). During evolution the simulated robot is connected to the (real) FPTA in the same way as the physical robot is when evolved controllers are used in the real world.

The original Heidelberg FPTA setup allows a maximum of seven concurrent buffered IOBs (Langeheine, 2005) which was not sufficient for our robotics experiments, especially when using vision. A workaround was developed which effectively allowed simultaneous use of all 64 IOBs: all inputs whose values are to be maintained must be associated to a common sample line at all times. In order to update specific inputs the corresponding inputs are moved to unique sample lines and then returned to the common line after. For reading outputs, each cell is configured to a unique sample line and then back to nothing. A maximum of four inputs can be updated, or six outputs read, simultaneously. This is due to three of the seven sample lines having a different channel implementation through the host FPGA (D_n as opposed to S_n in the darkgaqt GUI). This workaround incurs a performance penalty as I/O configuration is rewritten to the FPGA before every FPTA pin read/write. This is mitigated by keeping track of the current I/O configuration to optimise pin read/write scheduling. The maximum read write cycle through the FPTA interface is 2.2 Mhz. (See **Supplementary Material** for further details, available at https://www.mitpressjournals.org/doi/suppl/10.1162/evco_a_00272.)

2.3 The Robot

A K-Team K-Junior wheeled mobile robot (Lambercy, 2011) was used with a SmartEvo vision turret having a conical mirror mounted above its upward facing camera to provide 360° vision (Figure 4). The camera lens, focus and orientation were fixed such that the full panoramic image occupied as much of the camera field of view as possible (Figure 4). The K-Junior base was flashed to "slave mode" in order to communicate via the turret's fast Wi-Fi. The robot is circular with a 6.5 cm radius. Five of its six IR proximity sensors are distributed within a forward facing 180° arc at the front, and the sixth faces backwards at centre rear.

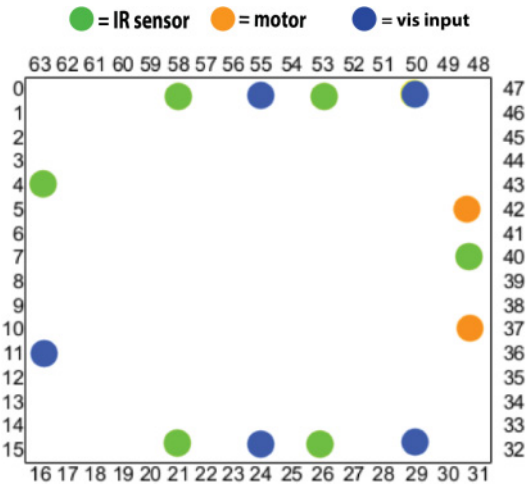

Figure 3: Layout of the FPTA IO pins and how they are mapped to the various robot sensors (6 IR sensors, 5 visual inputs) and 2 motor outputs.

Communication between the robot and a computer (which in turn communicates with the FPTA) was set up as follows (Figure 2): *IR and motor signals*: A robot interface socket server was written in C to run on the SmartEvo turret providing the K-Junior serial API over Wi-Fi while accessing native K-Junior library calls. *Vision*: an MPG streamer was started with 188 × 120 resolution at 10 frames a second and accessed via Wi-Fi from the computer.

The 188 × 120-resolution was used due to limitations of the SmartEvo turret camera which achieves less than one frame per second at its full 752 × 480 resolution. The camera's poor resolution, limited noisy colour range and slow rate made the visual tasks considerably more challenging, and provided the kind of test we desired to explore the capabilities of evolved transistor circuits to cope with low grade uncertain sensory data, as explained in the introduction. The proprietary camera turret communication system also introduced errors and noise into the reading of IR sensors and the writing of motor commands, which added to the challenges.

Figure 3 shows how the various robot sensors and motors are mapped to specific FPTA IO pins.

2.4 The Robot Arena

The robot arena used in all the experiments described later was a 85 × 114 cm open topped plywood box with various visual features stuck on the walls, most notably a red A4 sheet next to a black A4 sheet used as the goal during visual homing. A variety of obstacles were introduced within the arena during simulation and live tests.

2.5 The Robot Simulator

In order to evaluate robot behaviours in simulation, a carefully constructed simulation of the robot and its interactions with the environment was developed. The robot's kinematics and response of its IR sensors were modelled using techniques that have a long and successful track record at Sussex and elsewhere. The most complex part of the simulation—modelling vision—employed novel techniques developed for this research.

An efficient physics-based simulator was written in the style of Jakobi et al. (1995) to model the kinematics, assuming a flat floor. The velocity achieved for a given motor actuation value was derived empirically from a set of careful repeated measurements of the robot. A geometrically accurate model of the robot resolved movement into linear and rotational components, using the empirically derived motor signal versus wheel velocity response curve. Motor noise was introduced at a level empirically determined to match the real behaviour. Each IR sensor was modelled using an empirically derived response function which represented the way IR reflects off plain wood surfaces (measured repeatedly at many angles and distances), along with the underlying properties of the sensors (reflective spread, etc). Noise was introduced to the IR values at an empirically determined level. Each IR sensor was simulated by averaging beam reflectance over a spread of 5 traced rays 0.218 radians apart. A reflectance parameter could be altered to give a close approximation to reflective properties of materials other than wood. Motor and IR noise levels were also parametrized. The simulator operated at a configurable discrete time step δt.

The method used to simulate vision employs an empirical sampling technique made feasible by the use of a 360° field of view. The technique builds on a method previously utilized in Baddeley et al. (2011), but with significant extensions and improvements. As far as we know, it is the first time such a technique has been used in an evolutionary robotics context. The basic idea was to divide the whole world (the robot arena) into a set of equally sized cells. The image seen by the robot was then sampled in each grid cell to build up a database representing the robot's visual world. Because the robot has 360° vision, the panoramic image at a given location is essentially the same for any orientation of the robot, it has just been rotated. Hence, instead of having to sample at each location for many orientations, a small number of samples is sufficient. The retrieved image can be easily mathematically rotated to match the actual robot orientation. It is this trick, which relies on the rotational symmetry of the 360° image, that makes the technique feasible; otherwise the number of samples needed would become too large.

The arena was sampled by capturing the world as seen by a north and a south facing robot at every location on a grid of 5×5-cm cells. Multiple orientations were taken into account for slight variations in arena floor tilt and/or variations in the camera angle. Also, due to the camera being off centre on the robot, this provides a finer grained sampling of the arena on the y axis. At any given moment during simulation, a north or south orientation is chosen at random. The image sampled in that direction is picked from the sampling cell closest to the current position of the simulated camera. The chosen 360° image is then rotated according to the simulated robot orientation. The simulator also added noise to the sampled image (at empirically determined levels). The addition of noise and use of randomly chosen samples (north or south orientation) forces the evolved controllers to be robust to a range of visual conditions rather than relying on a fixed set of values. Such robustness is essential for transferring to the real world and operating in realistic conditions. The discrete nature of the sampling, and the use of the *nearest* sample to the actual position of the simulated robot, adds further noise and coarseness which increases the pressure to produce general, robust solutions.

In some of the experiments using vision, additional features and obstacles were introduced into the modelled arena through Computer Generated Imagery (CGI) injection into the sampled images. Hence the original sampled world can become the basis of new, more complex environments. In this work, such injected obstacles were limited

to vertical opaque cylinders with uniform colour reflectance at every angle. However, this aspect could fairly easily be generalised if desired.

2.6 Core Evolutionary Search Algorithm

A generational genetic algorithm was used with a binary genotype, linear rank selection, single point cross-over, mutation, and elitism. After preliminary investigations, a population size of 30 was used. With only a single FPTA which had to be used for each (expensive) evaluation this population size proved a good compromise, providing quick enough evolution. Preliminary experiments indicated cross-over probability of 0.6 and per bit mutation rate $0.001\overline{6}$ as the best values to use. The GA started with a population of 30 random genotypes. Each genotype was a fixed length binary string which encoded a FPTA configuration using 6144 bits describing transistor properties and connections for the array, as defined by the chip configuration protocol (Langeheine, 2005). This binary encoded protocol is hardwired into the chip design and *must* be used to configure the FPTA. Hence, it made sense to use it directly as the genetic encoding as intended by the chip designers (Langeheine, 2005) (any other encoding would have to be translated into it to enable chip configuration). A statistical asymmetry in the transistor channel length encoding defined by this protocol (010, 011, 110, 111 all mapping to 8 μm) was mitigated by finding redundant length encodings in the genotype and replacing them with random non-redundant ones. Preliminary experiments showed that this encoding worked well and so it was adopted for all FPTA experiments described in this article. (See **Supplementary Material** for full details of the encoding.)

For vision-based experiments an extra 120 bits were appended for visual sensor configurations (24 bits each for 5 evolved visual filters as described in Subsection 2.8).

An incremental approach to evolution (Harvey, 1992a) was used in the series of experiments described in this article. Task difficulty increased from stage to stage; each new stage was seeded with a population from the previous stage's final generation. Preliminary experiments indicated that this was the most efficient way to proceed as attempts to go straight to the final target behaviour failed; this is in keeping with previous explorations of this issue (Nolfi et al., 2016; Vargas et al., 2014).

2.7 Noise and Fitness Evaluation

Robotics is inherently noisy. Sensor and actuator noise and natural variations in environmental conditions (e.g., lighting) are always present (these are replicated in our simulations). In our case the physical medium used for the controller (unconventional FPTA circuits) can provide another source of inherent noise (e.g., parasitic capacitance build-up). Because of this, nominally identical fitness evaluations will result in different fitness values. Since we require the controllers to be robust to such variation, as well as to different initial conditions (e.g., position and orientation of the robot), the evaluation method must be carefully designed. Multiple trials must be used and these should be appropriately weighted in order to produce a selection pressure towards general and robust behaviours.

In the evolutionary robotics experiments detailed in the next section, N evaluations were made per individual and overall fitness was integrated using Equation 1. For most runs, $N = 6$. Each evaluation started from different randomized initial locations and orientations. The shape of noise deliberately introduced in the simulations was dictated by a random number generator (RNG) as in Garvie (2005). In order to rank individuals correctly it is important to provide all members of the population with the same generated noise so that they are all evaluated in the same set of conditions, helping to

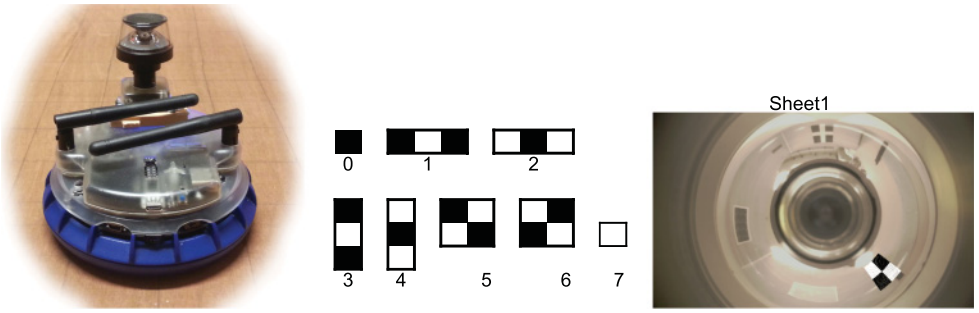

Figure 4: Left: the K-Junior robot with the SmartEvo turret having a conical 360° mirror mounted above the upward facing camera; Middle: Haar-like filters available as visual sensors; Right: example of filter ID 6 overlayed on the conical mirror 360° view. White regions are additive and black regions subtractive; see Subsection 2.8 for details.

ensure that their relative fitnesses (and hence rankings) are accurate. This makes sure the rank-based selection mechanism is not too noisy, which is important because rank determines the relative contributions of individuals to the next generation. It is done to help mitigate against some individuals getting much more favourable evaluation conditions than the others (e.g., all starting positions near the target) and hence receiving an artificially high relative fitness (and ranking) which will likely have a deleterious effect on the next generation. N RNG seeds were used per generation to create N sets of noise used for all individuals.

A simple mean fitness score could be used to integrate a set of fitness evaluations, f_i, into a final overall fitness value F. However, by giving a heavier weighting to lower fitness values as per Husbands et al. (2010), robustness is encouraged: the robot must behave well on *all* trials. Hence fitness function F (Equation 1) was used in all experiments. The scores, f_i, over a set of N evaluations are ranked (rank 1 is best (highest), rank N worst (lowest)). Thus F weights the evaluations scores inversely proportionally to fitness.

$$F = \frac{2}{N(N+1)} \sum_{i=1}^{N} i f_i \tag{1}$$

2.8 Vision: Processing and Genetic Representation

Here, we describe how vision was handled in our experimental setup. Instead of using the whole camera image as input to the controllers, five Haar-like feature detectors (or filters) (Viola and Jones, 2001) (whose position, size, and other properties are genetically set, as described later) were used to effectively evolve visual sensors which use only part of the image and pre-process it into a low dimensional input for the FPTA. This approach, where visual sensors are coevolved with the controller, builds on Harvey et al. (1994) and Husbands et al. (2010) but uses more sophisticated filters than in previous work. It is a powerful approach to automatically achieving dimensionality reduction and feature extraction and selection in an integrated way. Haar-like feature detectors calculate the averages and differences of intensities over adjacent rectangular regions (the black and white regions shown in Figure 4). They can act as simple edge and line detectors, as well as responding to more complex visual features and picking out areas of high contrast. A number of these feature detectors acting in concert have been shown

to work well in robot navigation, providing an efficient, low dimensional representation of visual data (Baddeley et al., 2011). The output, s (in the range [0,1]), of a Haar-like feature was calculated by overlaying the filter on the 360° conical mirror view as in Figure 4 (right) and applying the following equation:

$$s = \frac{\sum_{p \in W} v_p + \sum_{p \in B} (255 - v_p)}{255 \times (|W| + |B|)} + \Gamma \tag{2}$$

where W and B are white and black filter regions respectively, as in Figure 4, v_p (in the range [0,255]) is the value for pixel p on the channel this filter is genetically configured to, as described below, and Γ is uniform random noise in the range [0,0.1]. Areas of the visual sensors falling outside of the visual field are ignored.

The image quality provided by the robot SmartEvo vision turret is poor: it is very washed out and lacking in colour information (Figure 4, right). In order to mitigate this to some extent, camera images were white balanced using the "Grey World" algorithm (Finlayson et al., 1998) and extra hand-picked values of 40, 45, and 30 were subtracted from all rgb pixel values, respectively, for calibration of black. Two channels were then extracted from this rgb image:

- Greyscale: $0.2989r + 0.587g + 0.1140b$
- Red: $r \times MIN(1, \frac{r}{g+b})$, where b is clipped to a minimum of 10^{-8}.

The red channel responds most strongly to pure red and has low values for both white and black. In order to broaden the scope of possible evolved visual processing, it was genetically determined which of these channels any Haar-like feature detector used as raw input.

In order to allow evolution to operate in a very general, unconstrained way in the design of the visual filters, an extra 120 bits of the genotype encoded the configuration for the five Haar-like filters (24 bits for each), determining their sizes, positions, and other properties. Numerical values were represented using a Gray code (Gray, 1953) so that single bit mutations cause incremental changes in values resulting in a smoother fitness landscape. Each filter effectively acted as a separate visual sensor, feeding into its own FTPA input. The configuration encoding was as follows:

bits 1-3 filter type as per Figure 4 (middle)

bits 4-9 filter centre x: range [0,64] with 32 = straight ahead on panoramic image. Values increase clockwise; both 0 and 64 represent straight behind.

bits 10-14 filter centre y: 0 is at top, increasing values nearer to bottom.

bits 14-18 filter height such that the maximum value fills the full height of the field of view and 0 is a single pixel. Width is scaled such that each filter section is a square.

bits 19 channel type: 0 for Greyscale and 1 for Red.

bits 20-24 thresholding value t such that visual sensor output is clamped to 0 if $< \frac{t}{100}$.

The height of the field of view was constrained such that some of the floor was visible while nothing above the arena wall was available to the visual sensors as controllers might otherwise evolve to depend on information external to the arena which may be

unreliable when transferred to the real robot: external objects may move around, the whole arena may be moved, lights may be switched on or off and so on.

Once the thresholding function was applied to the output of a Haar-like filter, the result was amplified to a [0,5] V range for FPTA input.

3 Incremental Visual Target Finding Behaviours

In this section, we describe the core experiments carried out to explore the potential of our approach to evolving unconventional analog circuits for robot control. An incremental evolutionary approach was used to develop the following series of behaviours: obstacle avoidance, visual target approach in empty environment, visual target approach in complex cluttered environment. At each stage the previous behaviours were subsumed.

3.1 Obstacle Avoidance

This was the basic first-level behaviour on which the others were built; it ensured the robot didn't crash into the obstacles and the arena walls. The aim of this task was for the robot to cover as much ground as possible without colliding with obstacles (basically move in straight lines avoiding obstacles). FPTA pins 26, 21, 40, 58, 53, and 7 were allocated as inputs from the six robot IR sensors, with values normalized to the [0,5] V range. Values are high when the sensor is close to an obstacle. Vision was not used in this experiment. Pins 42 and 37 were allocated as outputs to the robot motor such that 0 V represented 20 (full forwards) and 5 V represented -20 (full backwards), with linear interpolation in-between. During initial experiments the simulated arena was a 150 $\times$ 150 square with bounding walls and an S shaped obstacle within it as in Figure 6 (left).

Fitness was calculated as in Floreano and Mondada (1994) and Jakobi et al. (1995):

$$f_a = \sum_{t=0}^{t=trialEnd} V(1-\sqrt{\Delta v})(1-i) \tag{3}$$

where V is the sum of the instantaneous rotation speed of the wheels (stimulating high speeds), Δv the absolute value of the algebraic difference between the speeds of the wheels (stimulating straight line forward movement), and i is the normalised value (in range [0,1]) of the IR sensor of highest activation (stimulating obstacle avoidance). This simple behaviour has been achieved many times with various ER/EHW approaches and is not in itself particularly interesting. However, it is a good basic test of the FPTA approach and is necessary as an initial bootstrapping behaviour in our incremental methodology.

The robot starts at position (30 cm, -15 cm) from the centre of the arena facing in a random orientation. Each of 6 trials ends when 200 simulated seconds is reached or the robot crashes into an obstacle, whichever is sooner. The simulation time step δt is 200 ms. The same sets of N IR, motor and orientation noise shapes are used to evaluate all individuals in a generation (see Subsection 2.7).

In live robot performance evaluations (see Subsection 2.3), the maximum I/O loop frequency was found to vary around a mean of about 20 Hz but was limited to a steady 10 Hz to allow rigorous comparisons and replications.

3.2 Homing: Vision-Based Goal Approach in an Empty Arena

The second task in the sequence was evolved on top of the obstacle avoidance behaviour by starting from the final population of the previous task. The aim of this task was for

the robot to approach a target which it can only recognize by using vision (a red 29-cm × 21-cm rectangle next to a black 29-cm × 21-cm rectangle on one of the walls; see Figure 7). Other visual features (patches of colour or pattern) were stuck to the arena walls and could potentially be used for orientation, but also had to be discriminated from the target. At this stage the evolution of visual sensors was switched on as outlined in Subsection 2.8. The approach used builds upon (Harvey et al., 1994) and (Baddeley et al., 2011). FPTA pins 26, 21, 40, 58, 53, and 4 were allocated to IR values, pins 50, 29, 24, 55, 11 to five evolved visual sensors and 42, 37 to motor output. The skylights over the actual arena were covered so as to avoid major disruptions in lighting (often resulting in lens flaring) between the time of sampling and any live test. The robot was to approach the target from a random starting position and orientation in an otherwise empty arena without crashing into the walls (i.e., maintaining the obstacle avoidance behaviour alongside the visual target finding).

Behaviour was evaluated over a number of trials (6), each starting from a random location and orientation in the arena. Fitness was calculated as follows:

$$f_g = \frac{\sum_{\forall t.d_t \geq 10.5} (1 - \frac{d_t}{d_1}) + \sum_{\forall t.d_t < 10.5} 2}{t_{end}} \tag{4}$$

where d_t is the robot's distance to the target at time-step t, and t increases from 1 until the trial ends at t_{end} due to robot collision or timeout, whichever comes sooner. Double scores are awarded on every time-step the robot spends near the target without colliding with it. The simulated time-step δt was increased to 0.6s to account for increased vision latency on the real robot. Maximum trial length was 80 time-steps. Versions of this behaviour have been achieved before (Harvey et al., 1994) but here it is used as an incremental step towards the next stage more complex behaviour. It is also a good validation of the coevolution of visual sensors approach and is the first time an EHW approach has been used for a visual behaviour (of course evolved FPTAs have never previously been used for robotics tasks).

The live robot (see Subsection 2.3) maximum I/O loop frequency was about 5 Hz with a video stream latency of around 600 ms. No artificial delays were added.

3.3 Vision-Based Goal Approach in a Cluttered "Maze" Environment

The third task in the sequence was the same as the second but in a complex cluttered, maze-like environment such that from many locations—more than 50% of the arena—the target was hidden (see Figure 8). It was evolved on top of the initial target-finding behaviour by starting from the final population of the previous task. This was considerably harder than the empty arena task as a more general visual searching strategy had to be evolved in order to robustly find the target from any starting position in the environment. Cylindrical obstacles of varying colours were injected into the sampled arena images using CGI techniques (see Subsection 2.5, and Figure 8). The injected cylinders having the same colours as the real paper and cardboard obstacles (Figure 10), with ±20% uniform noise added. The x and y centres of the obstacles were also randomised by ±3 cm for every evaluation using the usual noise shaping (see Subsection 2.7) to discourage solutions depending on precise obstacle locations or colours. Preliminary experiments on transferring evolved controllers to the real robot revealed the following problems:

A bug in the proprietary K-Junior & Evo stack resulted in extremely low front IR sensor readings; these being more relevant when approaching cylinders than flat walls. A 600 ms video latency in the SmartEvo system (even at the lowest camera resolution)

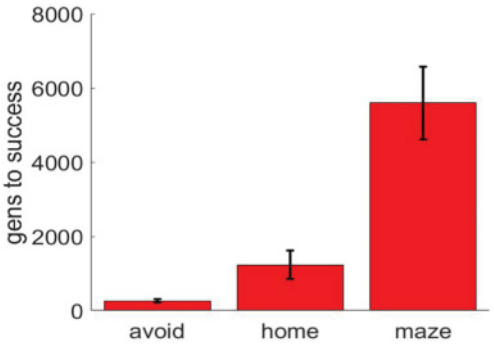

Figure 5: Results of ten incremental evolutionary runs. The bar heights show the mean number of generation to a successful robust solution at each stage; error bars indicate standard error of the means (SEMs).

caused the robot to miss landmarks it would otherwise approach: by the time the controller was "aware" of the landmark the robot had already turned past it and would see something different in the next frame. Motor values below 5 did not make the wheels turn with the Evo turret on the robot. Hence, evolved behaviour in which the simulated robot would slowly explore its surroundings, resulted in a stationary live robot.

All the robot shortcomings identified above were included in a modified version of the simulator which was then used in all the experiments reported here. Specifically, the time step δt was set to 0.3 s, simulated video latency to 600 ms (two frames delay), signals from the front IR sensor was clamped to 0 and motor values of -4 to 4 resulting in no wheel movement.

The evaluation procedure was exactly the same as in the previous task, except the maximum trial length was 200 time-steps. This task is more difficult than most previous ER visual behaviours and has not been attempted before (Nolfi et al., 2016).

4 Results of Incremental Evolution

Figure 5 (left) shows the summary statistics for ten incremental evolutionary runs. The height of the bars shows the mean numbers of generation to a robust, successful solution at each stage (that is, a high-scoring solution that remains the best in the population for 30 generations with re-evaluations on each generation, thus eliminating "lucky" individuals that scored well once; a high score for obstacle avoidance is 500, for both the visual tasks it is 1). The error bars show the standard error of the mean. All runs were successful at each stage, with moderate standard errors, indicating that the methodology is highly robust. All evolutionary runs started from fully random "primordial soup" populations except when seeded from a previous incremental evolutionary stage; that is, no hardcoding of genotypes was made at any point. Videos of all the behaviours described in this section can be found at www.sussex.ac.uk/easy/research/ehw-vids. At each stage very good transference to the real robot was achieved.

4.1 Obstacle Avoidance

After approximately 250 generations (mean $\pm$ s.e. $= 261 \pm 51$) optimal avoiders evolved on each run, going full steam ahead as much as possible. A typical such avoider is illustrated in Figure 6. This one had a slight bias towards a constant left turn, which helped

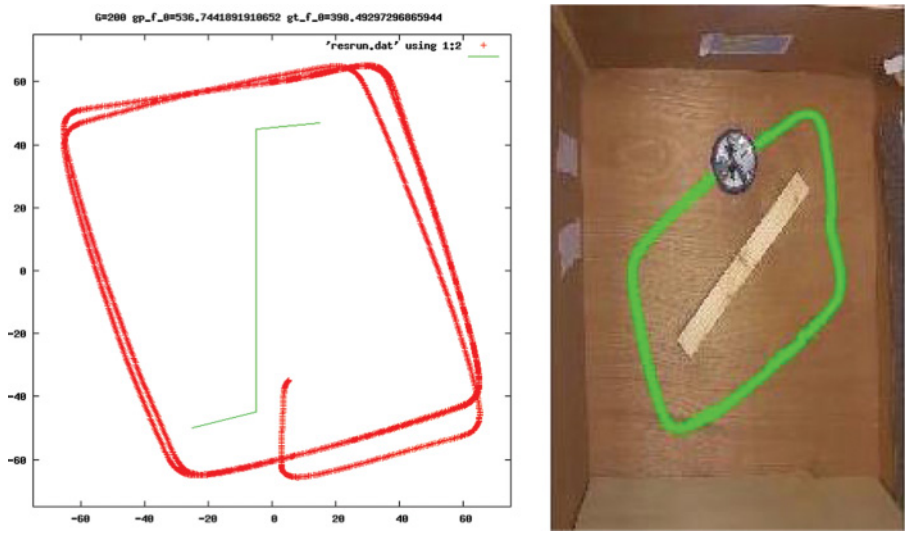

Figure 6: Obstacle avoider trace of fittest controller in a typical run, in simulator (left) and in real arena (right). The real robot trace was produced by processing an overhead video of the robot behaviour with motion tracking algorithms.

it lock into the optimal attractor around the central obstacle. When approaching a wall head-on, it usually turned left. Some others had a corresponding right bias, these were the two classes of successful avoiders that emerged.

The controller shown in Figure 6 transferred very well onto the real robot, producing qualitatively almost identical behaviour to that in the simulator, as can be seen on the right hand side of the figure. Ten fitness evaluations of this controller (the best at the end of an evolutionary run) were measured in the real world with the evaluation function used during evolution. Its mean fitness was 95.1 $\pm$1.6% of the mean fitness in simulation over ten evaluations. The robot was slightly slower in some areas of the arena, where the floor was not perfectly flat, and sometimes rotated away from walls more slowly than in simulation, which accounts for the slightly lower overall fitness. It generalized very well, dealing with obstacles introduced in real time and coping with environments more complex and cluttered than the one used for evolution. When approaching a wedge-shape corridor it slowed down and inched through the gap if it was wide enough, or stopped if not. (See videos at www.sussex.ac.uk/easy/research/ehw-vids.) All other successful avoiders transferred to reality equally well.

4.2 Vision-Based Goal Approach in Empty Arena

Seeded from the best obstacle avoider from the end of first stage, after approximately a further 1200 generations (1228 $\pm$ 381) robust successful visual goal finding (homing) controllers were evolved on each of the incremental runs. The behaviour of a typical successful evolved controller is shown in Figure 7. It begins by rotating clockwise until the goal is behind it. If it misses "locking the target" in one rotation (due to the low frame rate and distant target), it slows its rotation speed down until it catches it. At that point it starts approaching the goal in reverse with a very slight left turn. This left turn causes it to occasionally "unlock" the target which prompts a return to the clockwise rotation whereon the robot quickly locks back onto the target again: this amounts to

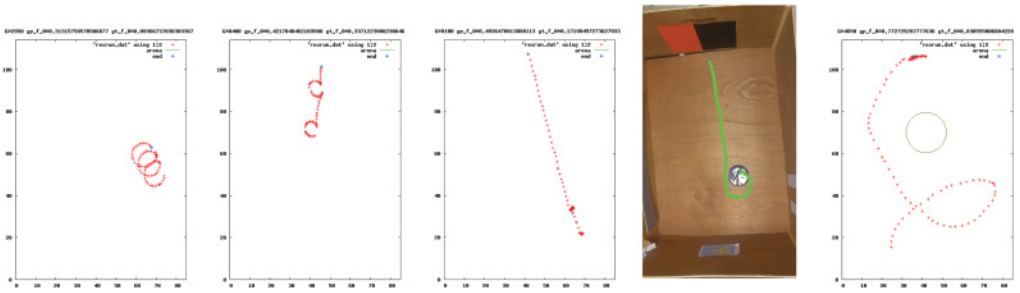

Figure 7: Traces of the fittest empty arena visual based goal approachers in simulation near the start (left), in the middle (second left), and at the end (mid) of a typical run (the target is in the middle of the top wall); second right: the final transferred solution in the arena showing the robot at its start position and its path to the target; right: the robot generalising to an unseen environment with an obstacle obscuring the view of the target.

an evolved calibration method which also allows it to deal with natural drift due to motor noise. Upon reaching the target, the robot maximizes its fitness score by moving back and forth from the target while staying within the (double scoring) goal zone: remarkably complex behaviour for considerably less than 256 transistors. Traces of the best individuals throughout this particular evolutionary run can be seen in Figure 7, giving an idea of how the behaviour developed over evolutionary time.

The best controller at the end of each evolutionary run transferred very well to the real arena (Figure 7) robustly identifying and approaching the target from all parts of the arena and qualitatively replicating the behaviour seen in the simulator. Ten fitness evaluations of the controller shown in Figure 7 were measured in the real world with the evaluation function used during evolution. Its mean fitness was $96.3 \pm 1.8\%$ of the mean fitness in simulation over ten evaluations. The robot was sometimes slightly slower in rotating and locking onto the target, and like the obstacle avoiders, was slower moving in some (unlevel) areas of the arena. Figure 7 (far right) shows the controller generalising to a fairly significant variation in the environment which was unseen during evolution, namely an obstacle obscuring the view of the target. The controller also generalized well to a moving "portable target" (a red plastic box) through its natural recalibration mechanism. On other runs the solutions found were very similar at the behavioural level; some rotated in the opposite direction, some moved forwards rather than backwards, some used slightly larger looping movements and tended to approach from more of an angle. The very fittest all produced behaviour very similar to that shown in Figure 7.

4.3 Visual-Based Goal Approach in Maze Environment

In each of the ten incremental runs, after approximately a further 5500 generations (5590 ± 986), seeded from the best individual from the previous stage, robust visual target finding behaviours evolved for a far more complex environment which included a maze in the arena in the form of cylinders of varying colours. Figure 8 shows plots of the best robot's paths to goal for a typical run. The robot approaches and stays at the target from any start position. With all visual sensors clamped to 0, this controller

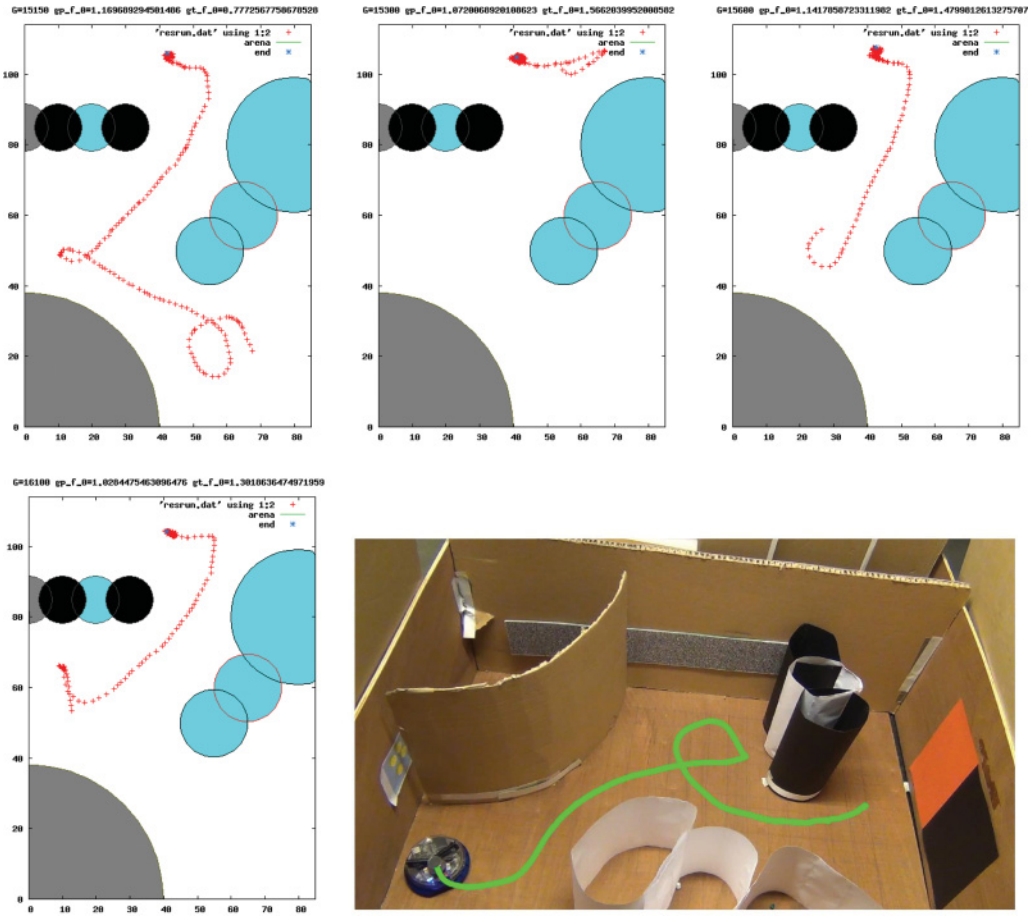

Figure 8: Traces of a typical evolved maze goal approacher from various starting locations and orientations in simulation and the final controller after transfer to the arena (bottom) with paper and cardboard cylinders. The initial (real) robot position is shown along with its path to the target. The quadrant containing the robot corresponds to bottom-right in the simulation traces and is referred to as such.

spins counterclockwise with a 12 cm radius whilst avoiding collisions when encountering a wall both on its left or right, demonstrating that it uses visual and IR sensing. An initial investigation of the controller revealed that when sensor #4 (greyscale light detecting sensor in the front-right of the robot's visual field in Figure 10) fires the robot straightens up. Given the location of the three interlocking white cylinders (appearing cyan and on the right in the simulated robot trajectories in Figure 8) and the size of the turning angle relative to the quadrants, the above is sufficient for the robot to arrive at the target reliably. The controller has adapted the robot's behavioural "morphology" or natural turning angle to the environment, and exploited efficient visual clues for navigation by using appropriate positions and properties for its visual sensors. As the robot reaches the target wall, it looses sight of the white cylinders and turns left, arriving with the target on its front right. Here the balance between sensor #1 and the others forces a clockwise turn, and the natural (and right IR induced) counterclockwise turn which

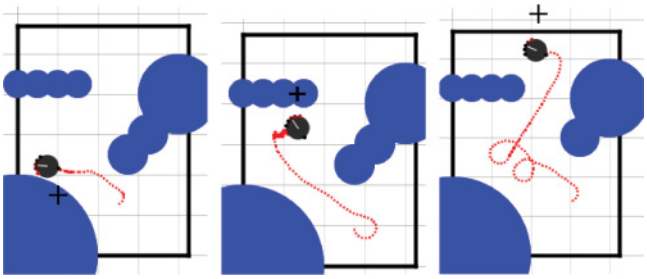

Figure 9: The evolved controller shown in Figure 8 generalising to environments unseen during evolution. The black cross in the behaviour plots indicates the location of the target on one of the walls/barriers. Left: target moved to lower barrier (during evolution it had always been on the top wall), middle: target moved to horizontal barrier, right: target in original position but part of the diagonal barrier removed to create variation in the shape of environment.

happens once it has rotated past the target, create a goal-sitting calibration mechanism. The amounts by which each wheel is made to go forwards or backwards given the target sensor and right IR values are precisely those required to accumulate maximum points during evolution by remaining in the goal zone. The results of this run with a complex environment is an example of evolution harnessing complexity (and asymmetry) in an environment in order to structure its behaviour and produce high fitness. By allowing evolution to shape and then exploit the robot-environment dynamics, along with the visual sensor morphology and properties, an elegant, and remarkably resource efficient, controller emerged. Although quite subtle, the evolved visual processing using to solve the task is extraordinarily minimal.

The evolved successful controllers transferred very well to the real robot, producing qualitatively very similar behaviour to the simulation; a trace of the live robot extracted from an overhead video can be seen in Figure 8. Ten fitness evaluations of the controller shown in Figure 7 were measured in the real world with the evaluation function used during evolution giving a mean fitness was 93.1 $\pm$2.1% of the mean fitness in simulation. Even though the controllers were evolved to produce very efficient behaviour in the environment shown in Figure 8 (with variation during evaluation trials as explained in Subsection 3.3), and exploited robot-environment dynamics particular to this environment, our methodology still made them general enough to be able to successfully perform the task (find the target and stay at it) in unseen variations of the environment where the target had been moved to a different location or the shape of the environment had been altered, as shown in Figure 9. This demonstrates that the successful controllers were processing sensory information to generate the behavior, rather than using some trick to blindly learn the shape of the environment and location of the target.

All in all, the limitations of a robot which can only move fast, and with a 600 ms camera latency and no front IR sensor (see Subsection 3.3), might appear damming and would lead many a conventional designer to lift their arms up in despair, especially when the task is navigating to a target in a complex environment using a maximum of 256 transistors. The success of our EHW approach shows the power of evolution to adapt under any circumstance having implications on robustness and adaptability for systems in the field. Evolution is neutral to "faulty" systems; it will strive to find a path towards a solution no matter what the medium.

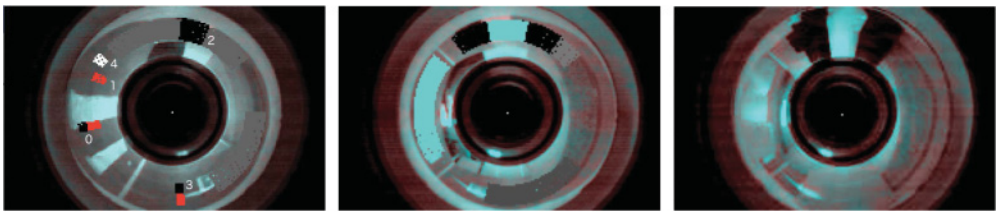

Figure 10: (left) Evolved visual sensors for maze homing controller overlaid over colour adjusted simulated robot view. Top is ahead, robot's right is image left. Sensor #4 plays a prominent role in navigation as does #1 in goal detection, the other visual sensors appear to have subtle modulating effects during orientation. (middle) Robot view from similar location in simulator and (right) real arena with colours adjusted to reflect two channel greyscale (as cyan) and pure red (as red) vision (see Subsection 2.8). Injected obstacles are rendered only within the robot's visual field (where sensors are allowed) resulting in squat quasi-1D vision when combined with full height evolved visual sensors.

5 Evolved Circuit Analysis

A circuit reduction and simplification methodology was developed to analyse the evolved circuits. This method proved very useful in giving a "first pass" understanding of the evolved circuits. Discrepancies between behaviours generated by the original and simplified controllers gave indirect evidence that the evolved controllers used active and subtle dynamics at the both the FPTA and robot-environment levels. (Full details can be found in the **Supplementary Material**.)

6 Comparative Evolutionary Runs

The results described in Section 4 suggest the FPTA is a suitably evolvable medium for developing robust sensorimotor behaviours, even when the sensors and motors are noisy and unreliable. The observed, rather subtle, dynamics of the evolved behaviours (especially the visually guided behaviours) suggest the potentially rich dynamics of the FPTA medium are being exploited. The analysis of evolved circuits (see Section 5 and **Supplementary Material**) further supports the exploitation of analog FPTA dynamics as being import. In this section this idea is explored further by comparing evolutionary results of the unconstrained FPTA with a setup where the FPTA dynamics are greatly constrained, and with other analog arrays that lack dynamics, as well as a system with rich dynamics.

It is possible to effectively bypass all the transistors in the FPTA by fixing all routing to be "pass through" so that the chip becomes an evolvable routing network. By doing this, the FPTA becomes a different kind of medium which can be used to evolve routing networks that can connect sensors and actuators in potentially complex (or relatively simple) ways but which no longer make any use of the transistors and the potential dynamics they can impart.

Figure 11 (top left) shows results from ten comparative incremental evolutionary runs of the full FPTA, the FPTA used as a routing network (transistors bypassed), and an array of simple threshold units. For maximum comparability, the latter was simulated and used the same 16×16 array layout as the FPTA, the same routing and input/output patterns and was evolved using the same genetic encoding as for the FPTA

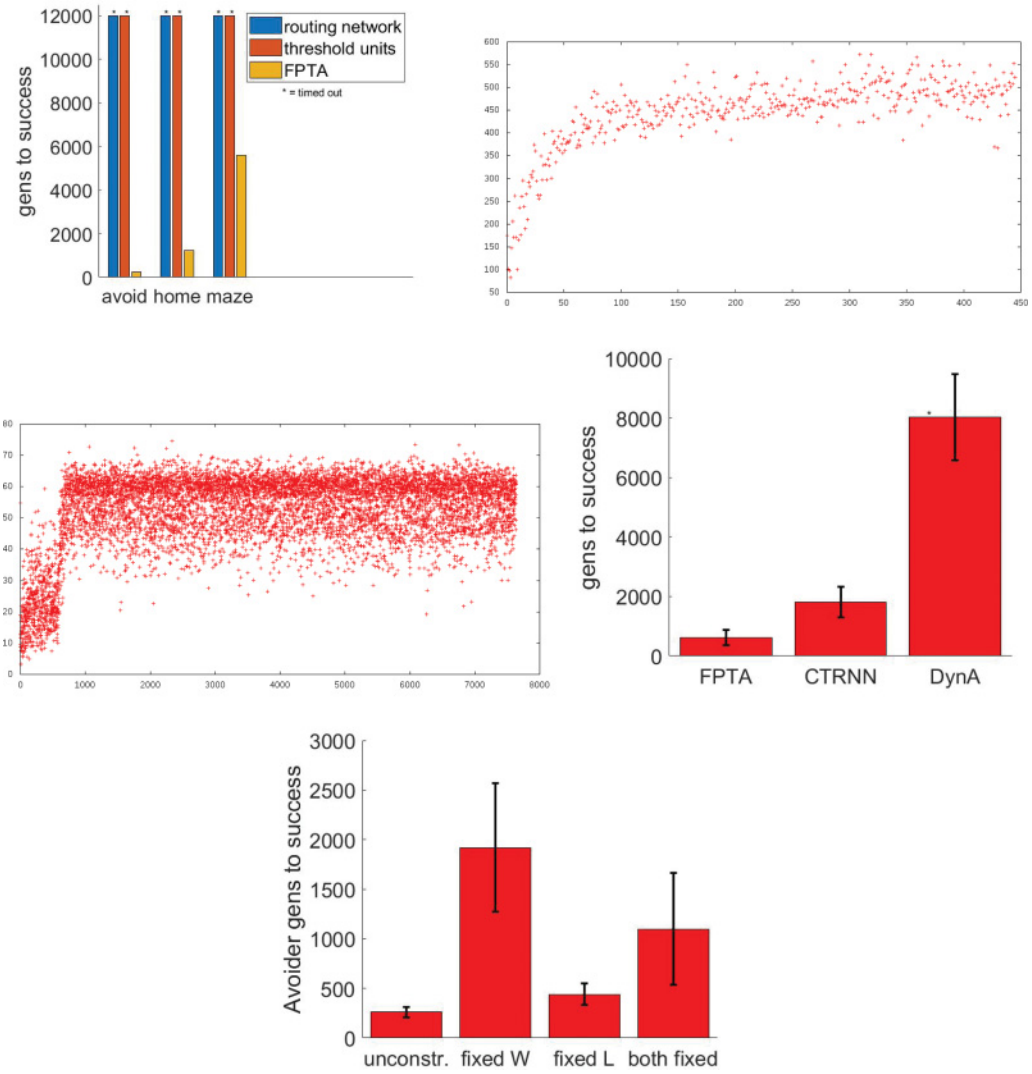

Figure 11: Results of comparative evolutionary runs. *Top left*: full FPTA compared with FPTA used as a routing network and with simple threshold units. An asterix above a bar indicates all runs under that condition timed out before success was achieved. *Top right*: best fitness vs generations for typical avoider run of FPTA with transistors active. *Middle left*: fitness plot with transistors bypassed. *Middle right*: FPTA compared on avoider behavior with array of dynamical units and with a dynamical neural network with a predefined architecture (CTRNN). *Bottom*: evolving avoider behaviour using different constraints on transistor properties, from unconstrained to fixed. The bar heights show the mean number of generation to a successful robust solution for each stage/condition; error bars indicate SEMs. Note the y-scale is different on each plot.

(except where the FPTA used two transistor properties per cell, width and length, the linear units use one—the threshold); these units are intended to resemble to some degree (highly idealised) transistors, without any dynamics. They operate according to

the following equation:

$$O = f(a, b) = \begin{cases} b & , a > T \\ 0 & , a \leq T \end{cases} \tag{5}$$

where a and b are inputs from two genetically determined neighbouring cells, O is the output of a unit and T is a genetically determined threshold (range: 0–1). *All* the full FPTA runs were successful at each stage. It is clear from the plots that *none* of the runs of the FPTA as routing network or of the array of threshold devices were successful at any stage. All of these runs times out at 12,000 generations. Multiple runs with the threshold devices using a fixed threshold of 0.5, or with evolved weights (range: 0–2) on the outputs were equally unsuccessful, as were runs with an array of different simple linear units (no longer resembling transistors, but still without dynamics) described by the following equation:

$$O = \begin{cases} min(k.I, 1) & , I > T \\ 0 & , I \leq T \end{cases} \tag{6}$$

where I is the total input to a unit, O is the output, T is an evolved threshold (range: 0–1), and k is an evolved gain (range: 0–2). Figure 11 (top right) shows plots of the highest fitness versus number of generations for typical evolutionary runs of the avoider behavior for the FPTA with transistors active, (middle left) shows the same for a typical run with the FPTA used as an evolvable routing network. While the full FPTA evolves very good solutions (fitness = 500) after about 250 generations, after about 1000 generations the routing network evolves (poor) controllers that charge forward and crash (fitness $\approx$ 60) and is not able to find any better solutions (i.e. ones that coordinate sensory input and motor output) after thousands more generations. Unsurprisingly, with such poor fittest individuals to seed the subsequent evolutionary stages in the overall incremental process, all of the homing and target finding in cluttered environment runs were also unsuccessful for both the routing FPTA and the threshold units.

These results, with the FPTA far superior to the other two systems, clearly indicate that the transistors in the array play an important role in the success of the FPTA as an evolvable medium for these kinds of tasks. Observations of the robot behaviours, which often exhibited quite subtle variations in speed, such as slowing down on a subsequent scanning movement when the target had been lost on the first pass, strongly suggested that internal dynamics were at play, presumably generated at least in part by interactions in evolved transistor circuits. To explore this further a new set of comparative runs was executed on the avoider behaviour. This time the FPTA was compared with units with explicit dynamics. Results of these experiments (ten runs of each) are shown in Figure 11 (middle right). An array of dynamical units (DynA in the figure) was created by replacing the threshold units in the simulated analog array setup of the experiments described above, with units defined by the following (leaky integrator) differential equation:

$$\tau \frac{dy}{dt} = -y + I \tag{7}$$

where τ is an (evolvable) time constant (range: [0.3,10]), y is the unit's "activation" and I its total input. A unit's output, O, is given by:

$$O = \frac{1}{1 + e^{-(y+\theta)}} \tag{8}$$

where θ is an evolvable bias (similar to a threshold, range: $[-5, 5]$). This form of equation is widely used in neural modelling and in dynamical neural networks employed

in robotics (Dayan and Abbot, 2005; Beer, 1997). The connectivity, biases and time constants for all units in the array were evolved using the same machinery and encoding as in the previous experiments. Out of 10 runs with the DynA setup, 5 were successful, the rest timed out after 12,000 generations. The DynA bar in the figure is a little misleading as it uses 12,000 generations for the unsuccessful runs to calculate the overall averages and standard errors, this is over generous but allows some kind of visual comparison with the other runs. Of the five successful runs, the mean number of generations to success was 4056 ± 1220. A fresh set of ten runs with the FPTA were all successful. The mean and standard error, shown in the plot, were similar to those found on the other earlier sets of runs, illustrating the reliability of the FPTA as an evolvable medium for noisy sensorimotor tasks. The partial success of the DynA experiments further suggested that dynamical capabilities were an important factor in success at this task, which is made relatively tricky by the noisy nature of the simulation (reflecting the noisy nature of the sensors and motors on the physical robot, and enabling robust transfer to reality as explained in Subsection 2.5).

To examine this further, a set of ten runs were performed which evolved a widely used kind of dynamical neural network (Beer and Williams, 2015) with a predefined architecture suitable for the task (labelled CTRNN in the bar plot). These runs moved away from the 16×16-array setup, which is directly comparable to the FPTA and in which control architectures have to be "carved out" by evolving suitable circuits, to the more constrained problem of setting the variables on a fixed neural network. It no longer made any sense to use the same genetic encoding as for all the other FPTA and array of threshold/dynamical units experiments. Instead, the genotype was an array of reals encoding the network variable as explained below. For comparability, the same form of evolutionary algorithm was used as with all previous experiments, except now the mutation operator was a random variation in a gene based on a normal distribution centred on the current value with std 0.2 (a so-called creep operator). A fixed architecture three layer 6-3-2 network was used. Each layer was fully connected to the previous layer (e.g., each node of the middle layer had connections from all of the nodes in the input layer). The six nodes of the input layer each took an input for one of the six IR sensors, the two nodes of the output layer provided the motor signals. The three nodes in the middle layer also had recurrent connections to themselves and each other. All connections had evolvable weights (range: $[-5, 5]$). Nodes in the network operate according to the following equations:

$$\tau_i \frac{dy_i}{dt} = -y_i + \sum_{j \in \Phi} w_{ji} \sigma(y_j + \theta_j) + I_i \tag{9}$$

$$\sigma(x) = \frac{1}{1 + e^{-x}} \tag{10}$$

where τ_i is a genetically set time constant for the ith node (range: [0.3,10]), y_i is the node's activation, Φ is the set of all nodes connected to node i, w_{ji} is the genetically set weight on the connection from node j to node i, θ_j is the genetically set bias for node j (range: $[-5, 5]$), and I_i is any external sensory input to node i. The summation is the total weighted input to node i from all other nodes connected to it (the output of a node is $\sigma(y + \theta)$). All ten CTRNN runs were successful. The high performance of this more elaborate and targeted architecture adds weight to our theory that the ability of evolution to manipulate internal controller dynamics is very useful in finding good solutions in this context. (See **Supplementary Material** for full details of the CTRNN setup.)

Although all of both the FPTA and CTRNN runs were successful (as defined at the start of Section 4), it can be seen from Figure 11 (middle right) that both the average and standard error of the FPTA runs (617 ± 264) is lower than that of the CTRNN runs (1806 ± 513). A non-parametric Mann–Whitney U test revealed that the FPTA is significantly better than the CTRNN at the 95% confidence level ($p = 0.049$). Clearly the DynA runs were much worse, half of them not completing successfully.

Having established that the transistors play a crucial role in the evolved FPTA controllers, and appear to make the medium suitably evolvable, a further set of comparative experiments were performed to get some insight into which, if any, of the transistor properties were most important during the evolutionary search in terms of evolvability, as measured by average speed to a good solution. Figure 11 (bottom) shows the results of evolutionary runs on the avoider task under different constraints on the way evolution could change the properties of the transistors in the FPTA. It shows ten runs under each of four conditions: no constraints, transistor width (W) fixed at the mid value, transistor length (L) fixed at the mid value, W and L both fixed at their mid values. All runs were successful. Pairwise Mann–Whitney U tests with the Bonferroni correction for multiple comparisons showed that the fixed W constraint is significantly less evolvable than both the fixed L constraint ($p = 0.03$) and the no-constraint condition ($p = 0.003$). Runs with both W and L fixed have a much higher average and standard error than the fixed L and unconstrained conditions but the difference in evolvability is only statistically significant at the 90% confidence level. The no-constraint condition has the lowest mean and standard error of the mean. These results suggest that the best approach is to allow evolution to control all transistor properties and routing connections (as was done in all other runs). The question remains: why does the fixed W constraint perform so much worse than the fixed L and unconstrained conditions, and have a much higher mean than the W and L both fixed runs? Fixed W runs get stuck in local optima much more often and for much longer. We believe this relates to the way the fitness landscape is determined by the chip configuration protocol which is built into the design of the chip (Langeheine, 2005) and which is used to map the binary genotypes to an FPTA circuit (see **Supplementary Material** for details). There are 16 widths to choose from, but only 5 lengths. Actual transistors use the same fixed 5 lengths, but width variation is accomplished by combining multiple transistors in the cell (i.e., setting up virtual transistors). So, for instance, changing from $L = 2$, $W = 1$ to $L = 2$, $W = 15$ means moving from a single $L = 2$, $W = 1$ transistor to four $L = 2$, $W = 1$, $W = 2$, $W = 4$, $W = 8$ transistors with their implied routing. This would likely cause a large behavioural variation given all the impedances of the routing and circuitry involved in hooking the transistors together. Hence, allowing W to vary provides a rich range of behavioural changes (some small, some big). There are far fewer possible values for L to take and the way the configuration protocol works means that single bit flips in the L encoding mostly results in relatively large changes, as opposed to the often gradual changes in W. Compared to changes in L, changes in W usually result in smoother progress from one parasitic (circuitry) combination to another. All this means that constraining W to a single value produces a less smooth, less correlated fitness landscape: it is better to allow W to vary. Fixing L has much less of an effect as long as W is unconstrained (indeed there is no statistical difference between the unconstrained and fixed L conditions).

7 Evolved Voice Control of the Robot

In order to investigate a wider set of sensory modalities, along with the integration of multiple evolved circuits, FPTA circuits capable of voice recognition were evolved. The

same basic evolutionary machinery, with the same parameter settings as used in the robotics experiments described above, was employed. This also allowed us to explore the applicability of the evolved FPTA approach to a different class of pattern recognition problems. The circuits were to identify the spoken words "Charge" and "Move." The evolved circuit was then integrated with a target finding circuit to allow voice control of the robot: on the command "Move" the robot is to move around the environment performing obstacle avoidance, on the command "Charge" the robot should switch to target finding (the target nominally marks the charging station). Training audio samples were captured within 10 cm of a laptop microphone with a pop shield through a custom audio capture tool. The whole set of audio samples was normalised to 98% peak volume and underwent Fast Fourier Transform (FFT) encoding into six frequency buckets such that the maximum possible bucket amplitude results in a value of 5 (to match with max 5 V FPTA input). Buckets f_0 to f_5 were further amplified by experimentally derived values 13, 11, 40, 16, 34, and 40 to encourage typical voice frequency amplitudes to fill the voltage range. Frequency buckets were fed into pins 28, 36, 56, 40, 50, 53 and output was read from pin 44. Samples were activated in a random order during evaluation with random gaps in between following a normal distribution $\mathcal{N}(10, 1.5)$. There was no added delay between read/write cycles. Output was expected to be 5 V for "Charge" and 0 V for "Move" samples. Fitness f was calculated using a reversed normalised sum of squared errors function:

$$f = 1 - \frac{\sum_{i=1}^{i=n} (D_i - Q_i)^2}{n \times 25} \tag{11}$$

where n is the total evaluation read/write cycle count (equal to the sum of samples and gaps), D_i and Q_i are desired and actual output at the ith evaluation cycle. For the word recognition task, word sample order variation does not have a large impact on fitness and inherent noise in the (physical) system dominates. This was mitigated by random FPTA reconfigurations between evaluations. Given that we cannot control the shape of inherent noise, some individuals might have "easier noise" than others. In the spirit of it being more important to rank individuals correctly rather than rushing to the next generation, during word recognition runs we start with $N = 3$ evaluations of an individual's fitness f and continue evaluating until we are 95% confident that f's true mean lies within a $\pm 5\%$ range around our current estimate. After each evaluation, the 95% confidence interval is calculated for the series of fitness values gathered so far. If this is greater than the 5% of the mean of this series we continue evaluating until a maximum $N = 25$ evaluations.

In a live setting, software automatically recorded audio samples above a certain amplitude threshold for chip input while chip output at 10-Hz refresh was monitored on a graphical output.

Fifty samples of Michael Garvie saying "Charge" and "Move" in varying tones of voice were used as the training set. In a typical run FPTA configuration evolved after 100 generations which matched the output well, as shown in Figure 12. Output voltage is between 4.5 and 5 V for "Charge" and between 0 and 0.5 V for "Move" in all but two cases—98% correct recognition. This configuration generalized well, responding correctly to $\approx$95% of live samples of Michael's voice. It was demonstrated in a live robot control setting by having the word recognition circuit trigger the activation of either a separate evolved obstacle avoidance controller (for "move") or a visual target finding controller (for "charge"); see video at www.sussex.ac.uk/easy/research/ehw-vids. The circuit also generalized well to other voices not used in training: when the commands

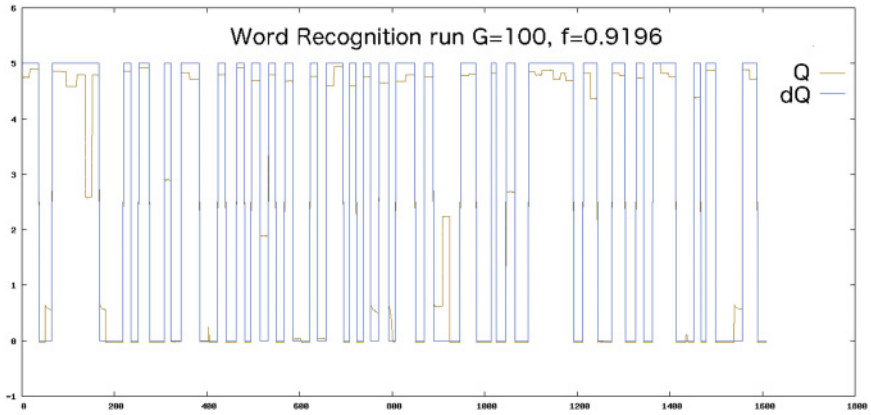

Figure 12: Example Word Recognition run with desired output dQ and FPTA output Q while being exposed to a random series of word samples.

were given by I. Flascher and P. Husbands, whose voices have very different tones (and accents) to each other and Garvie, the robot responded correctly. Similarly successful voice recognition circuits were evolved on every one of 12 repeated evolutionary runs, demonstrating the robustness of the approach.

Although tone discrimination circuits have previously been evolved (Thompson, 1998; Harding and Miller, 2004), this is the first time an EHW approach has been demonstrated to be capable of the more difficult task of word discrimination. For the purposes of the current article, we were interested in behaviour switching through the integration of multiple evolved circuits and multiple sensory modalities. This was successfully achieved as can be seen on the video mentioned above.

It should be noted that voice commands were simply used as a behavioural example of using the sound modality that required more complex processing of temporal features than simple tone discrimination, and was employed in order to further test the capabilities of the FPTA. Discriminating between two words in a fairly general way (as the FPTA managed) is not a difficult task for state of art word recognition techniques (which can discriminate between large numbers of words). However, those techniques are computationally very heavy. They typically employ multistage preprocessing of the sound waves with approximately 240 FFT bins followed by a bank of 20–30 Mel filter banks which usually generate Mel Frequency Cepstral Coefficients for 30 ms overlapping time frames (Deng and Li, 2013). The outputs of this are then fed into a machine learning system (often a large deep-learning neural network). What is interesting about the FPTA system evolved here is that it used only very minimal pre-processing of the sound wave: just 6 FFT bins, and was able to achieve the task in 100 generations.

8 Circuit Replication and Transfer

Early work in EHW revealed that circuits evolved direct in hardware (in practice on digital FPGAs) would generally not work as well, or at all, when transferred to another chip or another part of the same chip as it was evolved on (Thompson et al., 1996). This was because such circuits used physical particularities of the chip they evolved on, or the chip area they were evolved in. While demonstrating very interesting, highly unconventional designs, this was clearly a potential problem for practical applications of EHW.

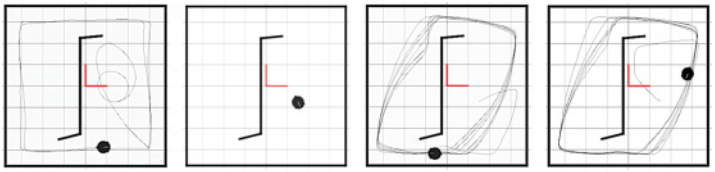

Figure 13: Traces of the best avoider evolved on the top half of the FPTA only when instantiated on the top (leftmost) and bottom (2nd left) halves of the chip, and an avoider evolved across both halves when instantiated on the top (2nd right) and bottom (rightmost) halves of the chip.

Later work explored similar techniques to those employed in evolutionary robotics to ensure greatly generality: namely evolving circuits in multiple conditions on each evaluation, which met with some success (Thompson and Layzell, 2000). An alternative approach, continuing/restarting evolution in situ for changed conditions/medium (e.g., to allow continued operation at higher temperatures), was also successful (Stoica et al., 2004).

A final set of preliminary experiments were conducted as an initial investigation into evolved circuit replication/transfer on the FPTA, in the context of robot control circuits. First, it aimed to see if there was a replication problem with the kind of FPTA medium used here, as this issue had not been investigated for it before. The goal was to evolve controllers independent of parametric particularities of specific chip components.

The task chosen was obstacle avoidance as in Subsection 3.1. In the first set of experiments FPTA configurations were limited to the top half of the chip during evolution using a bitmask whilst input and output pins were limited to the left and right edges of the upper half. IR sensor signals were fed into pins 40, 6, 43, 1, 47, 4 and motor outputs were read from pins 45 and 42 (see Figure 1 for positions of these pins). After evolution successful controllers were tested on the bottom half (which had not been seen during evolution) to check whether transfer would happen without any special evolutionary measures in place. After 300 generations a behaviour emerged whereby the robot turned on itself and slowly developed into an increasingly larger spiral until it performed straight line laps (see Figure 13, leftmost). At any point it would tend to turn left when facing an obstacle, resulting in counterclockwise arena laps. When moved to the unseen bottom half of the FPTA (mapping inputs and outputs to the matching pins on the bottom half) this controller simply spun on itself as per Figure 13 (second left), showing that it had indeed used specific properties of the top half of the FPTA, and special measures were needed to deal with transfer of evolved configurations to an unseen chip or part of chip. This experiment was repeated ten times, confirming the finding: on each run controllers were successful in the top half but failed in the bottom half; they did not generalise over the chip.

In the second part of the experiment, the controller was evolved with configuration settings for the top half, but evaluation was now performed *in both* halves of the chip. On each of its multiple fitness trials (as per Subsection 3.1) the controller was randomly (Subsection 2.7) either kept in the top half of the chip or transferred to the bottom half. Transfer was implemented by moving configuration bits to the second half of the array, keeping zeros in the first half, while I/O was moved to pins 32, 14, 35, 9, 39, 12 for IR and 37,34 for motor signals. After 500 generations an avoider similar to the one described

in Subsection 4.1 emerged, with a slight right turn resulting in a clockwise attractor round the arena. This controller worked exactly the same in both halves of the FPTA as shown in Figure 13 (two rightmost plots). Evaluating in both halves during evolution produced a pressure to generalise over the chip so that the controller would work well in either half and could not rely on specific particularities of one region. This demonstrates that evolving for generality of FPTA medium to secure replication and transfer of evolved circuits works well for intra-chip transfer in this context, and strongly suggests it would also work for inter-chip parametric variations. The same finding was repeated on ten runs of the experiment which all produced general controllers that worked in both halves of the chip.

The use of Equation 1 in evaluation provides a strong drive to generalization over all the mediums/environments used in evaluation, but the relative ease of achieving this result suggests the analog FPTA medium, with its rich, fluid dynamics, might be particularly amenable to this kind of generalization.

Because only one FPTA chip was available, the transfer/replication issue could not be fully investigated; that will be the subject of future work when more chips are available. Ideally the circuits would be evaluated over many chips/areas of chips/conditions during evolution and then tested on unseen chips/areas of chips to confirm generalisation.

9 Discussion

Component level analog electronics have been successfully evolved to act as controllers for a physical robot engaged in nontrivial visually guided behaviours. The evolutionary process successfully exploited the dynamics of the analog EHW chip, in conjunction with the robot–environment dynamics, to produce robust behaviour even with noisy, uncertain low-grade visual sensors, making use of an integrated evolutionary visual feature extraction and selection method. Thus both the questions posed in the introduction have been answered in the positive. We also demonstrated, for the first time, evolved word recognition circuits that were used in a behaviour switching task that integrated several evolved circuits.

Novel methods were developed to enable this work which has expanded FPTA-based EHW into the realm of realtime processing of sensor data coupled to actuator control in physical systems. A new method for simulating robot vision during evolution was presented, with excellent transfer to the real robot. The evolved solutions to non-trivial visual navigation tasks are best viewed as dynamical systems with (behavioral) attractors that result in completion of the task regardless of start conditions (Husbands et al., 1995; van Gelder, 1995). The continuous analog medium of the FPTA seems a particularly good substrate to enable the evolution of such attractors. This work supports the view of cognitive processes such as learning and memory as the reconfiguration of behavioral attractors in an embodied dynamical system (Beer, 1997; Beer and Williams, 2015). The reconfiguration of properties of the virtual transistors in each cell of the array are achieved by configuring combinations of actual (fixed property) transistors within the cell. Perhaps this process, with the addition routing and potential interactions within the cells, contributes to the rich dynamics of the FPTA medium and the level of evolvability we have demonstrated, particularly in comparison to the alternatives with limited dynamics we tried.

Our overall approach involved incremental evolution. Some interesting preliminary experiments were run to probe this approach further. In a simple version of the maze cylinder task (Section 3) a cylinder was inserted and removed from the arena

centre during evolution. Visual homing controller populations evolved with a cylinder historically present in previous generations, and since removed, were then much faster to adapt to its reintroduction, by displaying very efficient navigation around it, than populations solely evolved in an empty arena. This suggests that genetic material encoding the previous adaptation was still present in the population, ready to kick in, or at least the population remained in an area of fitness space from which it was easier to rediscover this adaptation. A fuller investigation of this effect, particularly on how to exploit it for more adaptable and evolvable systems, will be the subject of further work.

Because of the methodology we used, involving multiple fitness evaluations under noisy conditions, most evolved controllers generalized well to variations of the environment. However, for more radical changes in environment, some more specifically targeted evolution was used. For instance, in a separate experiment black markers were added around the simulated arena walls in order to facilitate transfer of homing to a lab floor with several dark objects around. The resulting controller transferred well to a complex environment, approaching a red box from up to 160 cm (limited by camera resolution).

An interesting phenomenon was observed in early word recognition runs. We found that after a while there was a sudden drop in population fitness. This was caused by some configurations performing better after repeated evaluations, most likely through exploitation of parasitic capacitance build-up. As long as the capacitance was maintained, such solutions would be acceptable when deployed in the field. However, as they came to dominate the evolving population, fitness increased until the evaluation of a new variation that discharged the capacitance resulted in a sudden drop in population fitness. We also found that other (unchanging) configurations could lose fitness after many evaluations, causing population fitness decline as they took over the population. Programming random FPTA chip reconfigurations between evaluations mitigated both scenarios and all solutions found after this measure was introduced were confirmed to retain fitness over long evaluations. This is a reminder that while many of the unconventional mechanisms exploited in EHW can results in highly robust efficient circuits, sometimes they need to be treated with care, and may have to be prevented from emerging.

Despite what turned out to be quite severe limitations in the robot when using the vision turret (high-latency, low-definition poor quality image, and nonoperational front IR sensor), we were able to reliably evolve robust, high performing controllers with our setup. This suggests that analog EHW might also be a useful approach for low cost realtime hardware applications requiring cheap sensors and simple circuits. However, in future work we will also investigate the use of the methodology with better robotic equipment. Having only a single FPTA chip was a considerable restriction. Multiple FPTAs would allow use of a fully distributed GA (Muhlenbein et al., 1991; Garvie, 2005) greatly increasing evolution speed and efficacy, as well as interchip transfer validation. A reconfigurable analog chip design with mid- and long-range routing, operational power and temperature monitoring, and on-the-fly self-reconfiguration would vastly open up behavioural solution space, whilst allowing parametric optimisation for specific applications. Such chips would enable a move to the next level of system: those capable of continuous adaptation and self organisation. Such systems have also been shown to be more evolvable (Husbands et al., 2010), so we believe the future of EHW lies in that direction.

Acknowledgments

Thanks to Martin Trefzer for help with initial setup of the FPTA, to Chris Johnson for help and discussion, and to the anonymous reviewers for helpful comments on an earlier draft of this article. This work was supported by Intel and EU ICT FET Open project INSIGHT.

References

Adamatzky, A. (2013). *Reaction diffusion automata: Phenomenology, localisations, computation*, volume 1 of *Emergence, complexity and computation*. New York: Springer.

Baddeley, B., Graham, P., Philippides, A., and Husbands, P. (2011). Holistic visual encoding of ant-like routes: Navigation without waypoints. *Adaptive Behavior*, 19(1):3–15.

Beer, R. (1997). The dynamics of adaptive behaviour: A research program. *Robotics and Autonomous Systems*, 20:257–289.

Beer, R., and Williams, P. (2015). Information processing and dynamics in minimally cognitive agents. *Cognitive Science*, 39:1–38.

Bekey, G. (2005). *Autonomous robots: From biological inspiration to implementation and control*. Cambridge, MA: MIT Press.

Berenson, D., Estevez, N., and Lipson, H. (2005). Hardware evolution of analog circuits for in-situ robotic fault-recovery. In *2005 NASA/DoD Conference on Evolvable Hardware*, pp. 12–19.

Bo, N., Deboeverie, F., Eldib, M., Guan, J., Xie, X., Niño, J., Haerenborgh, D. V., Slembrouck, M., de Velde, S. V., Steendam, H., Veelaert, P., Kleihorst, R., Aghajan, H., and Philips, W. (2014). Human mobility monitoring in very low resolution visual sensor network. *Sensors*, 14(11):20800–20824.

Cagnoni, S., editor (2009). *Evolutionary image analysis and signal processing*. New York: Springer.

Campos, P., Lawson, D., Bale, S., Walker, J., Trefzer, M., and Tyrrell, A. (2013). Overcoming faults using evolution on the panda architecture. In *Proceedings of the IEEE Congress on Evolutionary Computation (CEC'13)*, pp. 613–620.

Dayan, P., and Abbot, L. (2005). *Theoretical neuroscience*. Cambridge, MA: MIT Press.

Deng, L., and Li, X. (2013). Machine learning paradigms for speech recognition: An overview. *IEEE Transactions on Audio, Speech, and Language Processing*, 21(5):1060–1089.

Finlayson, G., Shiele, B., and Crowley, J. (1998). Comprehensive colour normalization. In *Proceedings of the 5th European Conference on Computer Vision*, pp. 475–490. Lecture Notes in Computer Science, Vol. 1407.

Floreano, D., and Mondada, F. (1994). Automatic creation of an autonomous agent: Genetic evolution of a neural-network driven robot. In *From Animals to Animats III: Proceedings of the Third International Conference on Simulation of Adaptive Behavior*, pp. 402–410.

Garvie, M. (2005). *Reliable electronics through artificial evolution*. PhD thesis, School of Cognitive and Computing Sciences.

Garvie, M., and Thompson, A. (2003). Evolution of self-diagnosing hardware. In *Proceedings of the 5th International Conference on Evolvable Systems: From Biology to Hardware*, pp. 238–248. Lecture Notes in Computer Science, Vol. 2606.

Gray, F. (1953). Pulse code communication. US Patent 2,632,058.

Haddow, P., and Tufte, G. (2000). An evolvable hardware FPGA for adaptive hardware. In *Proceedings of the IEEE Congress on Evolutionary Computation*, pp. 553–560.

Harding, S., and Miller, J. (2004). Evolution in materio: A tone discriminator in liquid crystal. In *Proceedings of the 2004 IEEE Congress on Evolutionary Computation*, Vol. 2, pp. 1800–1807.

Harvey, I. (1992a). Species Adaptation Genetic Algorithms: A basis for a continuing SAGA. In *Towards a Practice of Autonomous Systems: Proceedings of the 1st European Conference on Artificial Life*, pp. 346–354.

Harvey, I. (1992b). Untimed and misrepresented: Connectionism and the computer metaphor. Technical Report CSRP 245. School of Cognitive and Computing Sciences, University of Sussex.

Harvey, I., Husbands, P., and Cliff, D. (1994). Seeing the light: Artificial evolution, real vision. In *From Animals to Animats 3: Proceedings of the 3rd International Conference on Simulation of Adaptive Behaviour*, pp. 392–401.

Howard, G., Bull, L., de Lacy Costello, B., Gale, E., and Adamatzky, A. (2014). Evolving spiking networks with variable resistive memories. *Evolutionary Computation*, 22(1):79–103.

Husbands, P., Harvey, I., and Cliff, D. (1995). Circle in the round: State space attractors for evolved sighted robots. *Robotics and Autonomous Systems*, 15(1-2):83–106.

Husbands, P., Philippides, A., Vargas, P., Buckley, C., DiPaolo, E., and O'Shea, M. (2010). Spatial, temporal and modulatory factors affecting gasnet evolvability in a visually guided robotics task. *Complexity*, 16(2):35–44.

Jakobi, N. (1998). Evolutionary robotics and the radical envelope of noise hypothesis. *Adaptive Behaviour*, 6(2):326–368.

Jakobi, N., Husbands, P., and Harvey, I. (1995). Noise and the reality gap: The use of simulation in evolutionary robotics. In *Advances in Artificial Life: Proceedings of the 3rd European Conference on Artificial Life*, pp. 704–720. Lecture Notes in Artificial Intelligence, Vol. 929.

Keymeulen, D., Durantez, M., Konaka, K., Kuniyoshi, Y., and Higuchi, T. (1996). An evolutionary robot navigation system using a gate-level evolvable hardware. In *Proceeding of the First International Conference on Evolvable Systems: From Biology to Hardware*, pp. 195–210.

Koza, J., Keane, M., Streeter, M., Mydlowec, W., Yu, J., and Lanza, G. (2003). *Genetic programming IV: Routine human-competitive machine intelligence*. Dordrecht: Kluwer Academic Publishers.

Koza, J. R., Andre, D., Bennett III, F. H., et al. (1996). Evolution of a low-distortion, low-bias 60 decibel op amp with good frequency generalization using genetic programming. In *Late Breaking Papers at the Genetic Programming 1996 Conference*, pp. 94–100.

Lambercy, F. (2011). *K-junior user manual v1.2*. Technical Report. K-Team, Yverdon-les-Bains, Switzerland.

Langeheine, J. (2005). *Intrinsic hardware evolution on the transistor level*. PhD thesis, Kirchhoff Institute for Physics.

Langeheine, J., Folling, S., Meier, K., and Schemmel, J. (2000). Towards a silicon primordial soup: A fast approach to hardware evolution with a VLSI transistor array. In *Proceedings of the 3rd International Conference on Evolvable Systems: From Biology to Hardware*, pp. 123–132. Lecture Notes in Computer Science, Vol. 1801.

Langeheine, J., Meier, K., and Schemmel, J. (2002). Intrinsic evolution of quasi DC solutions for transistor level analog electronic circuits using a CMOS FPTA chip. In *Proceedings of the 2002 NASA/DoD Conference on Evolvable Hardware*, pp. 76–85.

Layzell, P. (1999). Reducing hardware evolution's dependency on FPGAs. In *Proceedings of the 7th International Conference on Microelectronics for Neural, Fuzzy and Bio-Inspired Systems*, pp. 171–178.

Lohn, J., Kraus, W., and Colombano, S. (2001). Evolutionary optimization of yagi-uda antennas. In *Proceedings of the 4th International Conference on Evolvable Systems*, pp. 236–243.

Lohn, J. D., and Hornby, G. S. (2006). Evolvable hardware: Using evolutionary computation to design and optimize hardware systems. *IEEE Computational Intelligence Magazine*, 1(1):19–27.

Lopez, B., Valverde, J., de La Torre, E., and Riesgo, T. (2014). Power-aware multi-objective evolvable hardware system on an fpga. In *Proceedings of the 2014 NASA/ESA Conference on Adaptive Hardware and Systems*, pp. 61–68.

Mayol, W., Tordoff, B., and Murray, D. (2002). Wearable visual robots. *Personal and Ubiquitous Computing*, 6:37–48.

Miller, J., Harding, S., and Tufte, G. (2014). Evolution-in-materio: Evolving computation in materials. *Evolutionary Intelligence*, 7(1):49–67.

Mohid, M., Miller, J., Harding, S., Tufte, G., Massey, M., and Petty, M. (2016). Evolution-in-materio: Solving computational problems using carbon nanotube-polymer composites. *Soft Computing*, 20(8):3007–3022.

Muhlenbein, H., Schomisch, M., and Born, J. (1991). The parallel genetic algorithm as function optimizer. *Parallel Computing*, 17:619–632.

Naito, T., Odagiri, R., Matsunaga, Y., Tanifuji, M., and Murase, K. (1996). Genetic evolution of a logic circuit which controls an autonomous mobile robot. In *Proceeding of the First International Conference on Evolvable Systems: From Biology to Hardware*, pp. 210–219.

Nolfi, S., Bongard, J., Floreano, D., and Husbands, P. (2016). Evolutionary robotics. In B. Siciliano and O. Khatib (Eds.), *Springer handbook of robotics*, 2nd ed., pp. 2035–2067, Berlin: Springer.

Ping, Z., Rui, Y., and Junjie, D. (2017). Design of self-repairing control circuit for brushless dc motor based on evolvable hardware. In *Proceedings of the 2017 NASA/ESA Conference on Adaptive Hardware and Systems*, pp. 214–220.

Roggen, D., Hofmann, S., Thoma, Y., and Floreano, D. (2003). Hardware spiking neural network with run-time reconfigurable connectivity in an autonomous robot. In *Proceedings of the 2003 NASA/DOD Conference on Evolvable Hardware*, pp. 189–198.

Sakanashi, H., Iwata, M., and Higuchi, T. (2004). Evolvable hardware for lossless compression of very high resolution bi-level images. *IEE Proceedings—Computers and Digital Techniques*, 151(4):277–286.

Sekanina, L., and Zebulum, R. (2005). Intrinsic evolution of controllable oscillators in FPTA-2. In *Evolvable Systems: From Biology to Hardware, Proceedings of the 6th International Conference*, pp. 98–107. Lecture Notes in Computer Science, Vol. 3637.

Stoica, A., Keymeulen, D., Zebulum, R., Thakoor, A., Daud, T., Klimeck, G., Jin, Y., Tawel, R., and Duong, V. (2000). Evolution of analog circuits on field programmable transistor arrays. In *The Second NASA/DoD workshop on Evolvable Hardware*, pp. 99–108.

Stoica, A., Zebulum, R., Keymeulen, D., Ferguson, M., Duong, V., and Guo, X. (2004). Evolvable hardware techniques for on-chip automated reconfiguration of programmable devices. *Soft Computing*, 8(5):354–365.

Stoica, A., Zebulum, R., Keymeulen, D., Tawel, R., Daud, T., and Thakoor, A. (2001). Reconfigurable VLSI architectures for evolvable hardware: From experimental field programmable transistor arrays to evolution-oriented chips. *IEEE Transactions on VLSI Systems*, 9(1):227–232.

Thompson, A. (1995). Evolving electronic robot controllers that exploit hardware resources. In *Proceedings 3rd European Conference on Artificial Life (ECAL'95)*, pp. 640–656, Berlin. Springer.

Thompson, A. (1998). *Hardware evolution: Automatic design of electronic circuits in reconfigurable hardware by artificial evolution*. Distinguished dissertation series. New York: Springer-Verlag.

Thompson, A., Harvey, I., and Husbands, P. (1996). Unconstrained evolution and hard consequences. In *Towards Evolvable Hardware: The Evolutionary Engineering Approach*, pp. 136–165. Springer-Verlag. Lecture Notes in Computer Science, Vol. 1062.

Thompson, A., and Layzell, P. (2000). Evolution of robustness in an electronics design. In *Proceedings of the 3rd International Conference on Evolvable Systems*, pp. 218–228. Lecture Notes in Computer Science, Vol. 1801.

Thompson, A., Layzell, P., and Zebulum, R. (1999). Explorations in design space: Unconventional electronics design through artificial evolution. *IEEE Transactions on Evolutionary Computation*, 3(3):167–196.

Trefzer, M. and Tyrrell, A. (2015). *Evolvable hardware: From practice to application*. Natural Computing Series. Berlin: Springer.

Trefzer, M. A., Lawson, D. M. R., Bale, S. J., Walker, J. A., and Tyrrell, A. M. (2017). Hierarchical strategies for efficient fault recovery on the reconfigurable panda device. *IEEE Transactions on Computers*, 66(6):930–945.

van Gelder, T. (1995). What might cognition be if not computation? *Journal of Philosophy*, 92(7):345–38 1.

Vargas, P., DiPaolo, E., Harvey, I., and Husbands, P. (Eds.) (2014). *The horizons of evolutionary robotics*. Cambridge, MA: MIT Press.

Viola, P., and Jones, M. (2001). Rapid object detection using a boosted cascade of simple features. In *Proceedings IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Vol. 1, pp. 511–518.

Yang, C., Virk, G., and Yang, H. (Eds.) (2017). *Wearable sensors and robots*, Vol. 399 of Lecture Notes in Electrical Engineering. Berlin: Springer.

Yasunaga, M., Nakamura, T., Yoshihara, I., and Kim, J. (2000). Genetic algorithm-based methodology for pattern recognition hardware. In *Proceedings of the 3rd International Conference on Evolvable Systems: From Biology to Hardware*, pp. 264–173. Lecture Notes in Computer Science, Vol. 1801.

Zebulum, R., Vellasco, M., and Pacheco, M. (1998). Analog circuits evolution in extrinsic and intrinsic modes. In *Proceedings of the 2nd International Conference on Evolvable Systems*, pp. 154–165.

Supplementary Material for: Evolved Transistor Array Robot Controllers

M. Garvie, I. Flascher, A. Philippides, A. Thompson, P.Husbands

1 FPTA Configuration Genetic Encoding

The digital genome representing a FPTA configuration is 6144 bits long and uses the original Heiderlberg Langeheine (2005) configuration string mapping. There are 16×16 cells defined in raster order each using 24 bits as follows:

0..2 L (length)
- **000** 0.6
- **001** 1
- **010** 8
- **100** 2
- **101** 4
- **rest** 8

3..5 Gate terminal
- **000** E
- **001** N
- **010** Vdd
- **100** S
- **101** Gnd
- **110** W
- **rest** None

6..8 Source terminal
- **000** E
- **001** Vdd
- **010** N [sic]
- **100** S
- **101** Gnd
- **110** W
- **rest** None

9..11 Drain terminal
- **000** E
- **001** S
- **010** N
- **011** W
- **100** Vdd
- **101** Gnd
- **rest** None

12,13 Not used

14 $W+=8$ (width)

15 WE (routing on/off)

16 NW

17 NE

18 $W+=1$

19 $W+=4$

20 $W+=2$

21 NS

22 SE

23 SW

where N, S, E, W denote the north, south, east and west routing points for a transistor terminal; WE, NW, NE, NS, SE, SW define whether these cell pass routes are enabled (eg. WE is from the cell's west to east routing point); and transistor length L and width W are measured in microns and the longest length bias is evident by values 010, 011, 110 and 111 at bits $0..2$ all resulting in $L = 8\mu$m.

2 Simultaneous Use of all 64 IOBs

The original Heidelberg FPTA setup allows a maximum of 7 concurrent buffered IOBs Langeheine (2005) which was not sufficient for robotics, especially when using vision. A workaround was found experimentally which allows simultaneous use of all 64 IOBs: all chip inputs whose values are to be maintained must be associated to a common sample line (eg S_0) at all times. In order to update specific inputs the corresponding inputs are moved to different unique sample

lines (eg. S_1, S_2) and then returned to the common line after. For reading outputs, each cell is configured to a unique sample line (eg. S_1, S_2, D_0, D_1) and then back to nothing. A maximum of four inputs can be updated, or six outputs read, simultaneously. This is due to three of the seven sample lines having a different channel implementation through the host FPGA (D_n as opposed to S_n in the darkgaqt GUI). This workaround incurs a performance penalty as I/O configuration is rewritten to the controller FPGA before every FPTA pin read/write. This is mitigated by keeping track of current I/O configuration to optimise pin read/write scheduling.

The workaround was tested by configuring all FPTA cells to connect all transistor terminals to Gnd and enable WE (west to east) routing. Then all 16 left border IOBs (IDs 0..16) were configured to unique incrementing voltages using the technique above. Then the voltages at the 16 right border IOBs (IDs 47..32) were read and confirmed to be identical (allowing for some resistive loss) to those set on the left hand side. This would be impossible using the standard 7 concurrent buffered IOBs that the Heiderlberg darkgaqt software allows. Indeed their software would only allow such a simultaneous test across two FPTA rows (IOBs 0,1 as input and 47,46 as outputs). Having proved the workaround allows simultaneous use of 32 IOBs, and given its method, there is no reason why it would not work with all 64. However a maximum of 13 simultaneous IOBs were used for the robotics tasks presented in this work.

3 CTRNN Setup

The CTRNNs used in the comparative experiments employed the following node equations:

$$\tau_i \frac{dy_i}{dt} = -y_i + \sum_{j \in \Phi} w_{ji} \sigma(y_j + \theta_j) + I_i \tag{1}$$

$$\sigma(x) = \frac{1}{1 + e^{-x}} \tag{2}$$

where τ_i is a genetically set time constant for the ith node (range: $[0.3, 10]$), y_i is the node's activation, Φ is the set of all nodes connected to node i, w_{ji} is the genetically set weight on the connection from node j to node i, θ_j is the genetically set bias for node j (range: $[-5, 5]$), and I_i is any external sensory input to node i. The summation is the total weighted input to node i from all other nodes connected to it (the output of a node is $\sigma(y + \theta)$).

A fixed 6-3-2 (input layer-middle layer-output layer) architecture was used. Each layer was fully connected in feed forward fashion to the next layer, the middle layer was also fully connected (including recurrency) to itself.

The genotype was an array of 55 reals (encoding 33 connection weights, 11 biases and 11 time constants) with ranges fo reach gene as specified in the main paper text. The genotype was read from left to write with the first block of reals encoding the properties of the input nodes (bias, time constant and three connection weights per node, read in order), the second block the middle layer and so on.

The evolutionary search algorithm used the same basic framework as in the FPTA experiments (pop size= 30, elitism, linear rank-based selection). Probability of one point crossover = 0.6, probability of mutation = 0.036 per gene, mutation was a random variation in a gene based on a normal distribution centred on the current value with std 0.2.

The differential equation was integrated using the Euler forward method with a time step of 0.05 in order to determine y_i at each time step. Sensory inputs and motor outputs were updated in sync with the robot simulator.

4 Evolved Circuit Analysis

Because they do not follow standard constraints and do not conform to well understood design principles, analysing evolved circuits can be challenging (Thompson et al., 1999; Raichman et al., 2003; Milano and Nolfi, 2016). Ideally, component level circuit voltages would be used, but in the case of the FPTA chip employed in this work there are no facilities to measure voltages within the array circuits. However, the Heidelberg FPTA software netlist export function generates subcircuit definitions for each FPTA cell along with a top-level circuit connecting neighbouring cells, all suitable for use with an electronic circuits simulator. Hence the use of an industry standard circuit simulator becomes an attractive alternative. Use of such a simulator also allows the possibility of searching through vast numbers of possible 'active' circuits by swapping components in and out the circuit under investigation. Thus a methodology was developed to analyse circuits in simulation and then test out the hypotheses generated on the real FPTA.

The overall approach involved the automatic accumulation of circuit modifications (such as deactivating cells) while keeping the deviation of input/output behaviour within bounds, followed by the simplification of the 'active' circuits found in this way. The aim was to develop a useful technique that would give a 'first pass' understanding of the evolved circuits and provide a new method to add to the general circuit analysis toolbox. Analyses of two of the evolved circuits featured in the main paper (an obstacle avoider and visual target finder for the maze environment) are described below.

4.1 Circuit Analysis Methodology

FPTA configurations were analysed in the SPICE electronics simulator. Each cell subcircuit, generated by the netlist export function, consists of a single transistor with a width/length ratio representing the composite transistors selected by that cell's configuration. A cell subcircuit also contains resistors representing routing. Nonlinearity of routing switches is ignored. Python code was written to allow simulation of circuits having floating nodes by using a global simulation parameter to add a separate 10GΩ resistance between each node (not only those floating) and ground.

The first stage of analysis involved observing (motor) output voltages while ramping (sensor) input voltages up & down individually and in all pair combinations. This established relationships between input and output voltages which are used to represent the robot behaviour.

The second stage of analysis attempted to establish which FPTA resources were actively used to implement the circuit function. A program was written to select a cell at random, remove it from the netlist, and execute a simulation of output voltages as input voltages are varied as above. If the mean squared error (MSE) of the motor output voltages (compared to those measured from the original unaltered circuit) was too large then the selected cell was put back, otherwise the removal was confirmed and allowed to persist as the search process moved on to look for further cells to remove. Thus the largest set of cells that could be removed while maintaining acceptable performance was sought. Many runs were performed, and different approaches for the MSE acceptance criterion were tried such as a threshold that was hard, soft, or even dynamic. Generally MSEs of significantly less than $0.1V^2$ were used. Cells attempting to use a signal from a removed cell resulted in floating connections dealt with as above. The same approach was then used to clamp transistors either (preferentially) OFF or ON to see whether such further simplification could be performed while still maintaining the required input/output behaviour of the circuit as defined above. All OFF transistors were removed from the circuit without degrading performance. An empirical investigation discovered that ON transistors could be replaced by their differential resistance at an operating point of 4.0V scaled downwards by 7% while approximately maintaining the required circuit behaviour. After this stage of circuit analysis (which involves some simplifications and assumptions which will be

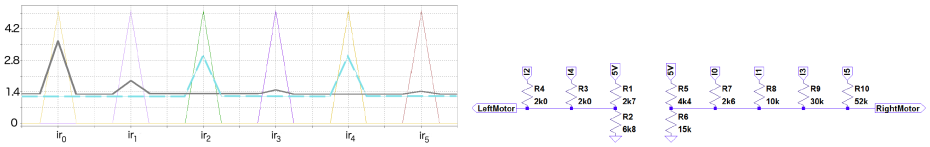

Figure 1: (left) Simulated avoider left motor (dashed) and right motor (thick solid) controller output voltage response to individual IR inputs (thin) ramping up to 5V and down.(right) Minimal reduced implementation of evolved obtacle avoider controller.

discussed later) the active circuit was reduced to a resistive network.

A third stage attempted further circuit simplification by automated application of network theory to the resistive networks. All resistances in parallel or in series can be combined without otherwise altering the topology. Further, such combinations may be found after applications of the invertible $Y - \Delta/\Delta - Y$ transform. Serial combination and the $Y - \Delta$ transform are special cases of the star-mesh transform which can eliminate any and all internal node(s), generally irreversibly O'Malley (2011). Custom software was written in Python to apply these operations to minimise those parts of a SPICE netlist that represent a purely resistive network.

4.2 Circuit Analysis: Obstacle Avoider

This analysis methodology was applied to the evolved obstacle avoider circuit described in §4.1 (main paper). It was analysed in SPICE yielding the I/O response to single inputs graphed in Fig.1: the right motor output increases (resulting in a rotation decrease and hence a right turn) with left IR sensitisation and left motor output increases with forward and rightmost IR input; as expected. However the baseline difference between motor outputs is opposite to that found on the FPTA and would result in a slight right turn in absence of inputs. Iterative random cell removal ended with 167 remaining cells. It was found that all transistors could be clamped ON or OFF and reduction of the resulting circuit resulted in a network of 10 resistors representing a straightforward Braitenberg-like avoider (Fig. 1).

Transistors were clamped in the same way on the actual FPTA to check the validity of this circuit reduction. The clamped circuit was used to control the simulated and actual robot. The behaviour observed captured the essence of the original behaviour generated by the unedited evolved circuit: successful obstacle avoidance was achieved. However, there were some marked differences: the robot moved slower, did not exhibit the optimal slight left turn to take a short route round the central obstacle, and occasionally reverted to a clockwise attractor round the arena due to its baseline slight right turn (interestingly also produced by the clamping) combined with a prevalent left turn when encountering an obstacle head-on. Hence, while the clamped controller produced successful obstacle avoidance the behaviour was less efficient than the original and it achieved significantly lower fitness during evaluation. Baseline controller output voltages in response to zero inputs are 0.77, 0.72 for the original and 1.37, 1.41 for the clamped version on the FPTA; and 1.18, 1.22 for the simulated unclamped circuit. Clearly the analysis has delivered some of the essentials of the evolved circuit, giving a strong first pass on how it operates, but it has not captured the full details and subtleties. These finding give indirect evidence that the evolved circuits use active dynamics discarded in the reduction method. Interaction dynamics in the controller-robot-environment system are also simplified and abstracted in the analysis which suggests they too play a part in the observed optimal behaviour in the unreduced circuit.

4.3 Circuit Analysis: Visual based goal approach in maze environment

The same analysis methodology was applied to the visual target finding controller described in §4.3 (main paper). Simulated I/O responses (Fig. 2) capture a core part of the robot be-

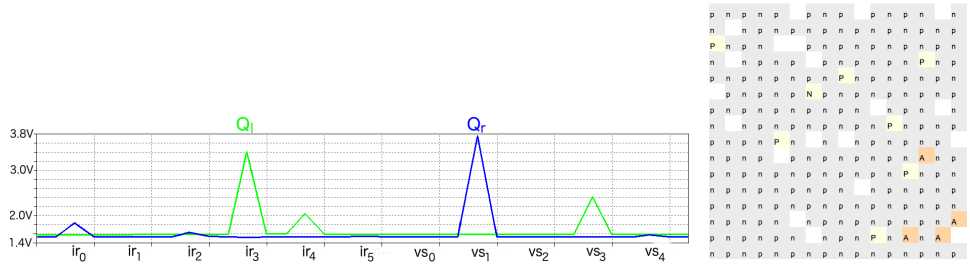

Figure 2: (left) Simulated maze homing controller motor output voltage response to individual IR and visual inputs (ramped as in Fig. 1), Q_l and Q_r are left and right motor outputs. (right) Maze homing controller FPTA schematic of cells clamped off (grey), on (yellow) and unclamped (orange).

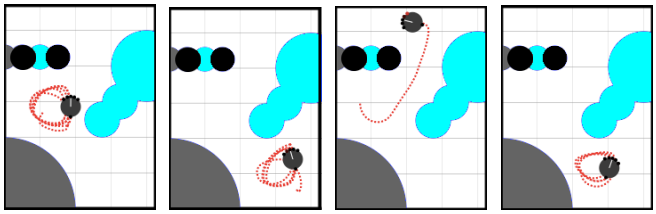

Figure 3: Traces of the resistive network synthesis (left two) and clamped FPTA controller (right two) exhibiting loss of function. Unclamped in Fig. 8 in main paper.

haviour: a lower right motor baseline directing a slight left turn in absence of input; visual sensor 4 resulting in equal motor outputs; strong responses to visual sensor 1 and IR sensor 3 which balance each other out in the goal zone behaviour. The iterative cell removal and transistor clamping processes left four transistors unclamped (Fig. 2 (left)). Resistive network simplification resulted in a network from which direct expressions for the motor outputs could be calculated as follows: left motor output $Q_l = (1.004016\text{e}{-2} + (ir_3/4.35e2) + (ir_4/1.82e3) + (vs_3/1.00e3))/6.287367\text{e}{-3}$; right motor output $Q_r = (2.192982\text{e}{-2} + (ir_0/1.20e3) + (ir_2/3.24e3) + (vs_0/2.19e4) + (vs_1/1.54e2) + (vs_4/9.49e3))/1.433231\text{e}{-2}$.

Clamping transistors on the actual FPTA in the manner suggested by the analysis (Fig. 2 (right)) resulted in a controller able to find the goal zone from all the arena except the bottom right quadrant (Fig. 3 (right)) whereas the resistive network controller only arrives at the target when it started in the top quadrant. Output for all three controllers was observed at the position from which the original leaves the bottom right quadrant: roughly at coordinates 57cm,29cm heading west. Input voltages are 0.09, 0.04, 0, 0, 0, 0, 0, 0, 3, 2.5, 4.35 with the original configuration outputting 1.79, 1.76; clamped 1.62, 1.41 and the resistive network 2.39, 1.57. These subtle differences are sufficient to result in loss of homing behaviour. Hence the reduced, clamped circuit appears to be a fairly good approximation of the actual evolved circuit, although its performance is not as good and it fails from the most difficult starting positions. The further simplifications inherent in the resistive network representation significantly degrade performance, only succeeding for the easiest starting positions. Clearly many of the subtleties and nuances of this more complex evolved circuit were lost during the further reduction stage, again suggesting it made use of active dynamics internally and at the robot-environment level.

4.4 Analysis Discussion

While it has proved to be a very useful tool, helping us to understand the evolved circuits by capturing their essence, there are inevitable simplifications and inaccuracies in the analysis methodology. Some shortcomings in the use of circuit simulations are evident from the obstacle avoider motor output baseline difference. These could be due to a number of causes: inaccu-

racy in the Heidelberg software 'spicify' function used to produce netlists (possibly including using nominal rather than accurately measured component values); floating node handling – we acknowledge that the addition of a 10GΩ resistance as outlined in §4.1 will not always replicate actual circuit behaviour; or parametric manufacturing peculiarities (we already observed a controller which worked on the top but not bottom half of the FPTA in §8, main paper).

The reduced circuits produced by the cell reduction method were found to work well in simulation and on the actual FPTA, although there was often some loss of performance on the real FPTA, particularly for the more complex behaviours such as that of the visual target finder. The transistor clamping method again worked quite well in terms of capturing the core of the evolved circuits but tests on the actual FPTA while controlling the robot revealed a degradation in performance with a loss of subtle dynamics (seen for instance in the slowing down and 'shuffling' behaviour original obstacle avoider when it approached difficult obstacles such as a narrow opening, or in the way the original visual target finder approached the target even from far away points in a complex cluttered environment). Reducing the circuits to resistive networks, although maintaining the core functionality for the simpler behaviours, resulted in further degradation of performance, particularly for the more complex robot behaviours.

There are a number of possible reasons for these discrepancies. The search processes involved in finding the necessary 'active' cells and in finding which transistors could be clamped, use an evaluation measure based on the MSE of the motor outputs for ramped sensor inputs. While this is fast and convenient and makes the technique viable, it is not quite the same thing as the actual robot behaviour. In particular, while efforts were made to make sure the measure held for simultaneous sensor inputs, it was primarily calculated with separated, single sensor inputs. In some circumstances, particularly for the visually guided behaviours, simultaneous sensory inputs appear to be involved in subtle non-linear dynamical behaviour. These may well not be covered by the simplified evaluation measure used in the analysis.

Another area of simplification that may account for some of the loss in dynamics in the reduced resistive networks is the replacement of ON transistors with differential resistance at a single fixed operating point. While this worked quite well for the simpler avoider circuits, it proved too gross a simplification for the more complex circuits with more subtle dynamics. Subtle transistor parametric changes – even very occasional ones – could cause significant behavioural differences when operating in the active range specially when in a cascade or feedback loop. Other behavioural differences could be due to parasitic capacitance (§9, main paper), cross-talk, and sequential effects.

Further refinements and developments of the analysis methods will be the subject of future work. This will include experimenting with in-situ versions of the iterative cell reduction and transistor clamping methods operating directly on the FPTA connected to the robot or robot simulator; white noise analysis on the FPTA, and, in the interest of arriving at minimal configurations, an extra parsimony fitness parameter during evolution as in (Garvie, 2005), which may result in circuits more amenable to analysis.

The general thrust of the analysis methodology (accumulation of modifications and simplifications) may be useful in other applications of evolutionary design where the issue of understanding the evolved structures is often a knotty problem.

References

Garvie, M. (2005). *Reliable Electronics through Artificial Evolution*. PhD thesis, School of Cognitive and Computing Sciences.

Langeheine, J. (2005). *Intrinsic Hardware Evolution on the Transistor Level*. PhD thesis, Kirchhoff Institute for Physics.

Milano, N. and Nolfi, S. (2016). Robustness to faults promotes evolvability: Insights from evolving digital circuits. *PLoS ONE*, 11(7):e0158627.

O'Malley, J. (2011). *Basic Circuit Analysis, 2nd Edition*. Schaum's Outlines. McGraw-Hill.

Raichman, N., Segev, R., and Ben-Jacob, E. (2003). Evolvable hardware:genetic search in a physical realm. *Physica A*, (326):265–285.

Thompson, A., Layzell, P., and Zebulum, R. (1999). Explorations in design space: Unconventional electronics design through artificial evolution. *IEEE Transactions on Evolutionary Computation*, 3(3):167–196.